Calhoun: The NPS Institutional Archive
DSpace Repository

Theses and Dissertations 1. Thesis and Dissertation Collection, all items

1991

Comparing combat models using analytical surrogates.

Green, John R.

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/26415

Downloaded from NPS Archive: Calhoun

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

http://www.nps.edu/library

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

Comparing Combat Models Using Analytical Surrogates

by

Captain John R. Green, U.S. Army

June, 1991

Thesis Advisor: Donald R. Barr

Approved for public release; distribution is unlimited

T256950

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b OFFICE SYMBOL *(If applicable)* 38	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS *(City, State, and ZIP Code)* Monterey, CA 93943-5000		7b. ADDRESS *(City, State, and ZIP Code)* Monterey, CA 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION	8b OFFICE SYMBOL *(If applicable)*	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c ADDRESS *(City, State, and ZIP Code)*

10 SOURCE OF FUNDING NUMBERS

Program Element No	Project No	Task No	Work Unit Accession Number

11 TITLE *(Include Security Classification)*
COMPARING COMBAT MODELS USING ANALYTICAL SURROGATES

12. PERSONAL AUTHOR(S) GREEN, JOHN R.

13a TYPE OF REPORT Master's Thesis	13b TIME COVERED From To	14 DATE OF REPORT *(year, month, day)* JUNE 1991	15 PAGE COUNT 91

16 SUPPLEMENTARY NOTATION
The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 COSATI CODES

FIELD	GROUP	SUBGROUP

18 SUBJECT TERMS *(continue on reverse if necessary and identify by block number)*
Combat Models, Simulated Annealing, Regression, Difference Equations, Stochastic Models

19 ABSTRACT *(continue on reverse if necessary and identify by block number)*

The widespread availability of inexpensive high-speed computers has led to the development of complex, detailed technical models of combat. These high resolution computer simulations and wargames are touted by their proponents as low-cost alternatices to extensive, high-cost field training exercises for the training of combat leaders. The validity of these simulations as models of combat, and thus as useful training tools is unproven. Direct comparison of simulations with field training exercises is often frustrated by the inherent complexities in each, and the shortage of quality data from field exercises. This thesis examines the feasibility of comparing these systems indirectly through the use of surrogate analytical models. A simple discrete time stochastic surrogate model is examined. Techniques for using the surrogate model to compare battle data are studied using simulated data from a simple combat model. Areas for further research are discussed.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS REPORT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL DONALD R. BARR	22b. TELEPHONE *(Include Area code)* 408 646-2763	22c. OFFICE SYMBOL MA/BA

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted
All other editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

Approved for public Release; distribution is unlimited.

Comparing Combat Models Using Analytical Surrogates

by

John R. Green
Captain, United States Army
B.S., Siena College, 1981

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN OPERATIONS RESEARCH
and
MASTERS OF SCIENCE IN APPLIED MATHEMATICS

from the

ABSTRACT

The widespread availability of inexpensive high-speed computers has led to the development of complex, detailed, technical models of combat. These high resolution computer simulations and wargames are touted by their proponents as low-cost alternatives to extensive, high-cost field training exercises for the training of combat leaders. The validity of these simulations as models of combat, and thus as useful training tools is unproven. Direct comparison of simulations with field training exercises is often frustrated by the inherent complexities in each, and the shortage of quality data from field exercises. This thesis examines the feasibility of comparing these systems indirectly through the use of surrogate analytical models. A discrete mathematical surrogate model is examined, and used to generate probability surfaces for comparison. We consider several techniques of fitting surrogate models to combat data, including simulated annealing, steepest descent, and multiple regression. Areas for further research are discussed.

T K6513
G728
c.1

THESIS DISCLAIMER

The Reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they can not be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

I. INTRODUCTION 1

II. METHODOLOGY 3

A. INTRODUCTION 3

B. ASSUMPTIONS 3

C. THE SURROGATE MODEL 4

D. LANCHESTER MODELS OF COMBAT 5

1. Non-Homogeneous Lanchester 13

2. Stochastic Lanchester 14

E. FITTING TECHNIQUES 17

F. MEASURES OF BATTLE 18

1. Pace of Battle 19

2. Relative Combat Power 19

3. Probability of Victory 19

4. Elasticity 20

5. Best Measures 20

G. COMPARING THE HIGH RESOLUTION MODELS 22

1. Methods of Comparison 22

2. Fighting Simulated Battles 23

3. Comparing Probability Surfaces 24

III. FITTING TECHNIQUES 29

A. INTRODUCTION 29

B. SIMULATED ANNEALING 29

C. STEEPEST DESCENT 32

D. NON-LINEAR OPTIMIZATION 33

E. LINEAR REGRESSION 34

F. CONCLUSION 35

IV. EXAMPLE OF METHOD 36

A. THE BATTLE DATA TRAJECTORIES 36

B. SELECT MATHEMATICAL SURROGATE SYSTEM 38

C. FITTING THE MODEL TO THE DATA 47

D. EMPIRICAL DISTRIBUTION FUNCTIONS 48

E. ANALYTICAL COMPARISON OF BATTLE DATA SETS 50

V. CONCLUSIONS AND RECOMMENDATIONS 53

A. INTRODUCTION 53

B. SURROGATE MODELS 53

C. FITTING TECHNIQUES 53

D. RECOMMENDATION 54

APPENDIX A: MATRIX SOLUTION TO DISCRETE MODEL 55

APPENDIX B: SIMULATED ANNEALING CODE 58

APPENDIX C: STEEPEST DESCENT CODE 63

APPENDIX D: GAMS REGRESSION MODEL 68

APPENDIX E: PROBABILITY SURFACE COMPARISON CODE 72

LIST OF REFERENCES 78

BIBLIOGRAPHY 80

INITIAL DISTRIBUTION LIST 81

I. INTRODUCTION

Comparing two high resolution models of combat has proven to be a difficult task. This thesis uses a simple discrete dynamical model of combat as a surrogate for the complex models being studied. Our surrogate model is fit to simulated battle data generated by a mathematical model, and used to generate probability surfaces. These surfaces are compared using a randomization test [Ref. 1] which proves effective at identifying battle data sets from similar simulations models and at distinguishing between battle data sets from different simulation models.

Land combat has evolved into a series of complex, combined arms battles fought with extremely expensive high-technology weapons. Many recent efforts to model this process have relied on computer simulations. These simulations attempt to capture the processes of combat by simulating the detailed interactions between individual combat elements (e.g. tanks, armored personnel carriers, and artillery pieces). As a result, these high resolution simulations are highly complex and expensive in their own right.

Having paid the price to develop these simulations, users rightly want to know if they replicate the combat processes modeled. This question has to a large extent gone unanswered. A great deal of data analysis, experimentation, and field testing goes into developing procedures used to simulate the various "microscopic" combat processes, such as searching for targets, identifying targets as hostile, and engaging hostile targets. Verifying that these procedures adequately model the processes for which they are designed is relatively straight forward. The interactions amongst all the various underlying processes are not well understood, and that makes the validation of the overall model difficult.

To validate a high resolution combat simulation such as Janus(A) [Ref. 2], an army battalion task force air-land combat simulation, one might try comparing some measure of effectiveness (MOE) from an actual combat engagement to the same measure of effectiveness for a simulation of that battle in Janus(A). However, not much data are available on actual engagements of high-tech armored forces, and actual combat data tends to be clouded by haphazard data reporting and collection techniques. The next best procedure might be comparison to data collected from realistic field training exercises such as those conducted at the United States Army's National Training Center (NTC) at Fort Irwin, California. The data from NTC, although collected more systematically than combat data, also has its shortcomings [Ref. 3], and it is expensive to collect. For these reasons it is difficult to get the desired number of replications of similar battles from NTC.

How then can we validate a model such as Janus(A) if there is no good source of data for comparison? This thesis attacks that problem by exploring ways to use analytical (mathematical) surrogate models of high resolution simulations, such as Janus(A) and field exercises at NTC, fitted to small battle data sets. Chapter II outlines the methodology proposed to compare two combat models. Chapter III examines the utility of simulated annealing, steepest descent, non-linear optimization, and multiple regression techniques for fitting the parameters of an analytical surrogate model to the simulation model's data. Chapter IV presents an example of the methodology to compare similar and dissimilar battle data sets generated from a mathematical model, and finally Chapter V concludes this study with a discussion of the utility of the methodology and proposals for further research.

II. METHODOLOGY

A. INTRODUCTION

This chapter examines the methodology developed for comparing two high resolution models of combat (referred to as simulation models throughout this thesis). First, we examine the assumptions necessary for our method, and the surrogate models considered (including a development of a discrete Lanchester model). We then discuss the techniques investigated for fitting battle data to the surrogate model and the measures considered for comparison. Finally, we examine the actual procedure used for comparing the two simulation models (i.e. two machine generated battle data sets representing simulation models).

Throughout this chapter we will refer to two critical items which are defined below.

- Battle trajectory. A battle trajectory is a numerical record of battle expressed as the number of each weapon system still active in the battle at the end of each time increment.
- Battle data set. A battle data set is a collection of battle trajectories from the same simulation model.

B. ASSUMPTIONS

Certain assumptions are necessary to implement this methodology.

1. Battle trajectory data are obtained as the raw number of each weapon system actively involved in the battle, recorded at the end of each time interval, as shown in Table 1; all time intervals are of the equal length.

2. It is assumed that each weapon system involved in the battle is capable of engaging any other weapon system involved in the battle. This assumption is required when using Lanchester-type equations as the surrogate model (Lanchester Models of combat are discussed in Section D).

TABLE 1. SAMPLE BATTLE TRAJECTORY.

Battle Trajectory from NTC					
Time	RTANK	RAPC	BTANK	BAPC	BTOW
0	60	80	50	50	10
5	56	73	47	46	9
10	52	67	45	42	8
15	48	61	43	39	7
20	45	55	41	36	6
25	42	50	39	33	5
30	39	45	37	30	4
35	36	41	35	28	3
40	34	37	34	26	2
45	32	33	33	24	1
50	30	29	32	22	0
55	28	26	31	20	0
60	26	23	30	18	0

3. The battle trajectories are assumed to be non-increasing (i.e. no reinforcements are allowed). This is often the case in computer simulation models. This assumption forces the combat processes' mathematical representation to be convex, thus simplifying the fitting of the analytical model.

4. Individual battle trajectories are assumed to be independent of all other battle trajectories. This is not a problem with computer simulations that are restarted using different random number seeds. However, this can be a problem with data obtained from field exercises such as those conducted at the NTC. Similar battles conducted by the same unit (or, even with the same opposing force) can display learning curve trends which could result in correlation between different battle trajectories.

C. THE SURROGATE MODEL

Choosing the surrogate model is a critical step in the analysis process. The model chosen must represent the underlying processes of interest. Ideally, the model should also be simple enough (i.e. have a small number of parameters to be estimated) to allow good fits to small data sets.

The model chosen for our initial trials was a discrete analog of the Lanchester aimed fire model. This model was chosen because it is an attrition model; because it has a small number of parameters to be estimated; and because the weapon systems considered were all direct fire weapons (i.e. tanks, armored personnel carriers, and anti-tank weapons). A short introduction to Lanchester-type models is provided in the next section.

D. LANCHESTER MODELS OF COMBAT

In the late eighteenth century Antoine-Henri Jomini wrote extensively on what he called the principles of war [Ref. 4]. One of these principles was that an army should mass (concentrate) its forces when attacking. Over a century later, in 1914, a British engineer and inventor F. W. Lanchester set out to prove a hypothesis similar to Jomini's [Ref. 5]. Lanchester conjectured that in "modern warfare" the concentration of forces was an effective tactic. To prove his hypothesis Lanchester developed a mathematical model of "modern warfare." Lanchester argued that with modern weapons it was possible for any combat element to engage any other on the battlefield, thus the attrition of a force should be proportional to the size of the opposing force. Lanchester presented the following model:

$$\frac{dX}{dt} = -aY; \quad a>0, \quad X(0) = x_0, \qquad (1)$$

$$\frac{dY}{dt} = -bX; \quad b>0, \quad Y(0) = y_0, \qquad (2)$$

where

$X(t) \equiv$ X force's strength at time t,

$Y(t) \equiv$ Y force's strength at time t [Ref. 6].

This model can be solved by dividing equation (1) by equation (2):

$$\frac{\frac{dX}{dt}}{\frac{dY}{dt}} = \frac{-aY}{-bX}, \qquad (3)$$

$$\frac{dX}{dY} = \frac{-aY}{-bX}, \qquad (4)$$

$$-bX\,dX = -aY\,dY. \qquad (5)$$

We then integrate both sides:

$$\int_0^t -bX\,dX = \int_0^t -aY\,dY, \qquad (6)$$

$$-\frac{b}{2}X^2\Big|_0^t = -\frac{a}{2}Y^2\Big|_0^t, \qquad (7)$$

$$-b(X^2(t) - X^2(0)) = -a(Y^2(t) - Y^2(0)). \qquad (8)$$

Thus we have:

$$b\left(X_0^2 - X^2(t)\right) = a\left(y_0^2 - Y^2(t)\right), \qquad (9)$$

which is known as Lanchester's Square Law.

Lanchester's model is a continuous approximation of the discrete combat attrition process. This formulation presents two problems for direct use as a surrogate model. First, in order to use Lanchester's model to simulate combat on a computer, it is necessary to approximate the differential equations with difference equations. Second, the observations we are using to estimate the parameters in our model are taken at discrete time intervals. It thus makes more intuitive sense to formulate the model with difference equations from the start. Lanchester's model expressed as a system of difference equations is:

$$X_{n+1} - X_n = -a\,Y_n; \quad a>0, \quad Y_0 = y_0, \tag{10}$$

$$Y_{n+1} - Y_n = -b\,X_n; \quad b>0, \quad X_0 = x_0. \tag{11}$$

By iterating equation (10) one step we have:

$$X_{n+2} - X_{n+1} = -a\,Y_{n+1}. \tag{12}$$

We then subtract equation (10) from equation (12) which yields:

$$X_{n+2} - 2\,X_{n+1} + X_n = -a\,(Y_{n+1} - Y_n), \tag{13}$$

and by substitution we get,

$$X_{n+2} - 2\,X_{n+1} + X_n = -a\,(-b\,X_n). \tag{14}$$

Thus,

$$X_{n+2} = 2\,X_{n+1} + (ab - 1)\,X_n, \tag{15}$$

which is a second order discrete dynamical system. It can be solved using standard techniques for second order difference equations [Ref. 7]. Its characteristic function is given by: $r^2 - 2r + (1 - ab) = 0$.

This equation has two real roots,

$$\begin{aligned} r &= \frac{-(-2) \pm \sqrt{4 - 4(1 - ab)}}{2} \\ &= 1 \pm \sqrt{ab}. \end{aligned} \tag{16}$$

Thus the general form of the solution to the discrete dynamical system for the X force is given by:

$$X_n = C_0(1 + \sqrt{ab})^n + C_1(1 - \sqrt{ab})^n, \tag{17}$$

where C_0 and C_1 are constants determined by the initial conditions. Applying our initial conditions:

$$X_0 = x_0 \qquad \text{and} \qquad X_1 - X_0 = -a\, y_0,$$

we obtain the following two equations in two unknowns:

$$X_0 = x_0 \qquad = C_0(1 + \sqrt{ab})^0 + C_1(1 - \sqrt{ab})^0, \tag{18}$$

$$X_1 = x_0 - a\, y_0 = C_0(1 + \sqrt{ab})^1 + C_1(1 - \sqrt{ab})^1. \tag{19}$$

Solving for C_0 and C_1 we get:

$$C_0 = \frac{x_0}{2} - \sqrt{\frac{a}{b}}\, y_0\, ,$$

$$C_1 = \frac{x_0}{2} + \sqrt{\frac{a}{b}}\, y_0\, .$$

Thus the particular form of the solution for the X force is:

$$X_n = \left[\frac{x_0}{2} - \sqrt{\frac{a}{b}}\, y_0\right](1 + \sqrt{ab})^n + \left[\frac{x_0}{2} + \sqrt{\frac{a}{b}}\, y_0\right](1 - \sqrt{ab})^n, \tag{20}$$

and similarly for the Y force:

$$Y_n = \left[\frac{y_0}{2} - \sqrt{\frac{b}{a}}\, x_0\right](1 + \sqrt{ab})^n + \left[\frac{y_0}{2} - \sqrt{\frac{b}{a}}\, x_0\right](1 - \sqrt{ab})^n. \tag{21}$$

The graph of these solutions with two different values of $\sqrt{ab}$ is shown in Figure 1. From this graph we see that the solution trajectories are approximately linear over the domain of interest (the first quadrant) with slopes determined by the value of

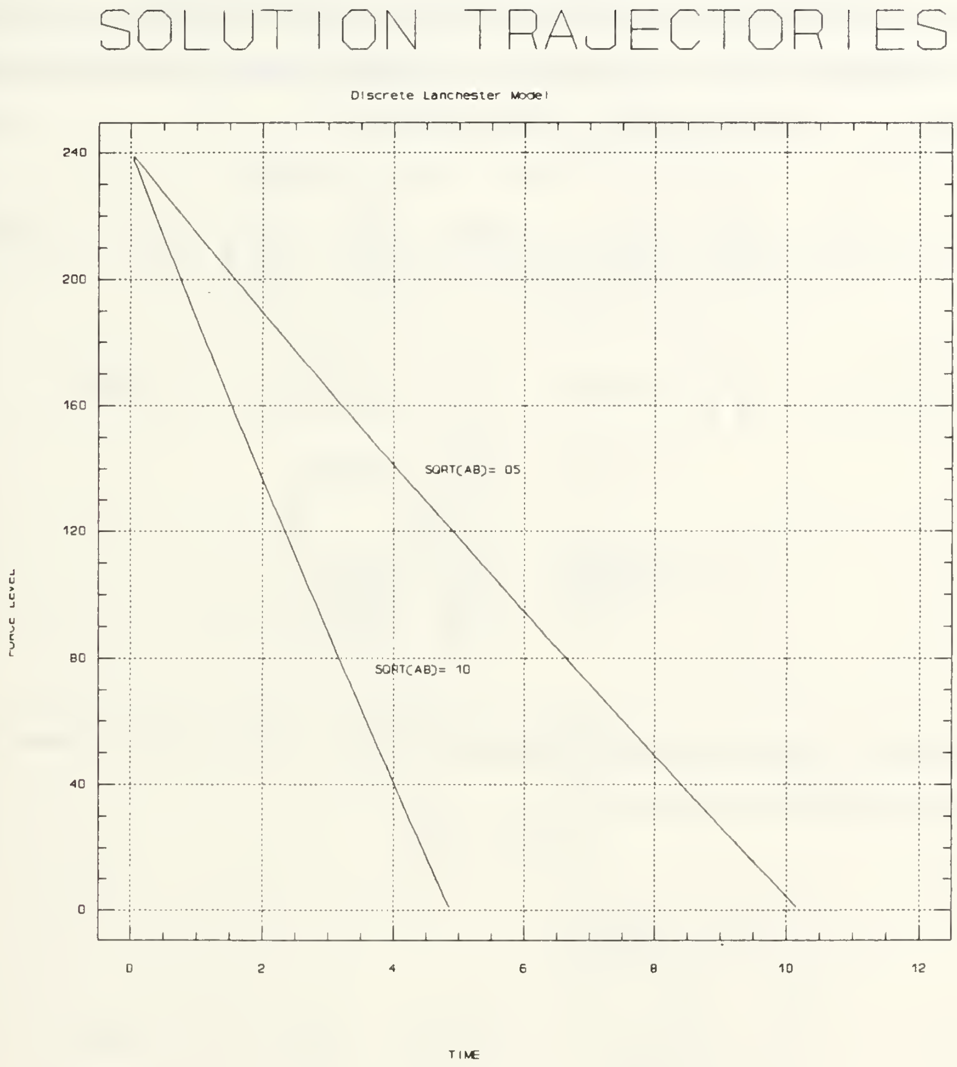

Figure 1. Solution Trajectories.

$\sqrt{ab}$. The larger the value of $\sqrt{ab}$ the more intense the attrition process and thus the steeper the slope. We can interpret the $\sqrt{ab}$ as a measure of the pace or intensity of battle (see Section F).

Another value of interest is the time at which the losing force would be annihilated. We are unable to calculate this value for the continuous Lanchester model because the solution trajectories go to zero asymptotically.

For the discrete model the annihilation time of the losing force is calculated by setting $X(n_x) = 0$ and solving for n_x. Thus we have:

$$X(n_x) = 0 = \left[\frac{x_0}{2} - \sqrt{\frac{a}{b}}\,y_0\right](1 + \sqrt{ab})^{n_x} + \left[\frac{x_0}{2} + \sqrt{\frac{a}{b}}\,y_0\right](1 - \sqrt{ab})^{n_x},$$

$$\left[\sqrt{\frac{a}{b}}\,y_0 - \frac{x_0}{2}\right](1+\sqrt{ab})^{n_x} = \left[\frac{x_0}{2} + \sqrt{\frac{a}{b}}\,y_0\right](1 - \sqrt{ab})^{n_x},$$

$$\left[\frac{1 - \sqrt{ab}}{1 + \sqrt{ab}}\right]^{n_x} = \frac{\sqrt{\frac{a}{b}}\,y_0 - \frac{x_0}{2}}{\frac{x_0}{2} + \sqrt{\frac{a}{b}}\,y_0}\,.$$

Now provided the right hand side of this equation is positive, which we show below, we can take logarithms on both sides:

$$n_x = \frac{\ln\left[\dfrac{\sqrt{\frac{a}{b}}\,y_0 - \frac{x_0}{2}}{\frac{x_0}{2} + \sqrt{\frac{a}{b}}\,y_0}\right]}{\ln\left[\dfrac{1 - \sqrt{ab}}{1 + \sqrt{ab}}\right]} \quad ; \quad \sqrt{\frac{a}{b}}\,y_0 - \frac{x_0}{2} > 0\,, \quad \sqrt{ab} < 1\,.$$

Similarly from equation (21) we get the annihilation time for the Y force:

We now show that the quantities in the above equations are positive. We know that if $\sqrt{ab} \geq 1$ then at least one of the attrition coefficients $a,b \geq 1$. This implies that in the battle, at least one of the forces would be annihilated in one time period. Thus we are only interested in the case when $\sqrt{ab} < 1$, and we have adjusted the length of our time periods to insure that $a,b < 1$. Thus in order to show that we can always determine the battle termination time n_x we need only show the following.

Lemma.

$$\textit{Assume } x_0,\ y_0 > 0 \ \textit{ and } \ 0 < a,b < 1.$$

$$\textit{If } \sqrt{\frac{a}{b}}\, y_0 - \frac{x_0}{2} < 0 \ \textit{ then } \sqrt{\frac{b}{a}}\, x_0 - \frac{y_0}{2} > 0\,.$$

Proof:

$$\textit{Assume } \sqrt{\frac{a}{b}}\, y_0 - \frac{x_0}{2} < 0\,.$$

$$\textit{Then } \sqrt{\frac{a}{b}}\, y_0 < \frac{x_0}{2},$$

so

$$\sqrt{\frac{b}{a}}\, x_0 > 2\, y_0 > \frac{y_0}{2},$$

$$\sqrt{\frac{b}{a}} - \frac{y_0}{2} > 0\,.$$

Q.E.D.

Using the formulas we have derived for the battle termination time we can now develop conditions for determining which force will be victorious. Setting $n_x = n_y$ we get:

$$\ln\left[\frac{\sqrt{\frac{a}{b}}\,y_0 - \frac{x_0}{2}}{\frac{x_0}{2} + \sqrt{\frac{a}{b}}\,y_0}\right] = \ln\left[\frac{\sqrt{\frac{b}{a}}\,x_0 - \frac{y_0}{2}}{\frac{y_0}{2} + \sqrt{\frac{b}{a}}\,x_0}\right],$$

$$\frac{\sqrt{\frac{a}{b}}\,y_0 - \frac{x_0}{2}}{\frac{x_0}{2} + \sqrt{\frac{a}{b}}\,y_0} = \frac{\sqrt{\frac{b}{a}}\,x_0 - \frac{y_0}{2}}{\frac{y_0}{2} + \sqrt{\frac{b}{a}}\,x_0},$$

so

$$\left(\sqrt{\frac{a}{b}}\,y_0 - \frac{x_0}{2}\right)\left(\frac{y_0}{2} + \sqrt{\frac{b}{a}}\,x_0\right) = \left(\frac{x_0}{2} + \sqrt{\frac{a}{b}}\,y_0\right)\left(\sqrt{\frac{b}{a}}\,x_0 - \frac{y_0}{2}\right).$$

Multiplying out the terms

$$\sqrt{\frac{a}{b}}\,\frac{y_0^2}{2} + x_0 y_0 - \frac{x_0 y_0}{4}\sqrt{\frac{b}{a}}\,\frac{x_0^2}{2} = \sqrt{\frac{b}{a}}\,x_0^2 - \frac{x_0 y_0}{4} + x_0 y_0 - \sqrt{\frac{a}{b}}\,y_0^2,$$

so

$$\sqrt{\frac{a}{b}}\,y_0^2 = \sqrt{\frac{b}{a}}\,x_0^2,$$

that is,

$$\frac{y_0^2}{x_0^2} = \frac{b}{a} .$$

$$\textit{Therefore if } \frac{y_0^2}{x_0^2} = \frac{b}{a} \textit{ then the battle is a stalemate,}$$

$$\textit{else if } \frac{y_0^2}{x_0^2} > \frac{b}{a} \textit{ then the Y force wins,}$$

$$\textit{else if } \frac{y_0^2}{x_0^2} < \frac{b}{a} \textit{ then the X force wins.}$$

This analysis of our model suggests a number of natural measures of battle which we will consider in Section F.

1. Non-Homogeneous Lanchester

Our model up to now has considered only homogeneous forces, such as a battle of tanks versus tanks. The models we wish to compare contain not only tanks, but armored personnel carriers and anti-tank weapons (they may also contain indirect fire weapons such a artillery; however, we will not consider them here). To include the differences in the attrition potential of the various weapon systems, we assign each weapon a weight, which we call its "fire power index." The total attrition potential of the X and Y forces is calculated as a linear combination of the individual fire power indices [Ref. 8].

$$Y = \sum_i f_{y_i} y_i ,$$

$$X = \sum_j f_{x_j} x_j , \qquad (22)$$

Where;

Y ≡ the total fire power of the Y force,

y_i ≡ the number of Y systems of type i,

and

f_{y_i} ≡ the fire power index of a Y system of type i,

and similarity for the X force.

2. Stochastic Lanchester

Our model has been deterministic thus far, with the battle results being fixed by the initial force levels on each side and the fixed attrition coefficients. There is intuitively much more to who wins a battle than simply the initial force levels. Leadership, training, momentum, and terrain are just a few of the critical factors in determining the outcome of any real battle. Introduction of these factors explicitly into our model, even if feasible, would make it too complex to fit to small data sets. We take a simpler approach by adding these factors as stochastic noise. Going back to the difference equation models, let us examine how this could be done. One simple approach is to add a random term to each equation:

$$X_{n+1} - X_n = -a\,Y_n + Z_x(n) ,$$

$$Y_{n+1} - Y_n = -b\,X_n + Z_y(n) , \qquad (23)$$

where Z_x and Z_y are random variables with given distributions whose parameters are to be estimated.

This approach, however, suggests that the noise is independent of the force sizes involved. This is intuitively unappealing as history shows the more complex the battle the more confusion likely to be present. We can incorporate this idea into our model by postulating that some factors, such as combat readiness and momentum, may be dependent on force size; while other factors, such as leadership and training may be independent of force size. Thus we separate these factors into two random terms:

$$\begin{aligned} X_{n+1} - X_n &= -A(n)\,Y_n + Z_x(n)\,, \\ Y_{n+1} - Y_n &= -B(n)\,X_n + Z_y(n)\,, \end{aligned} \qquad (24)$$

where A(n) and B(n) are random variables representing the force size dependent factors, and $Z_x(n)$ and $Z_y(n)$ are random variables representing the force size independent factors.

Further we assume that these factors are predetermined at some level at the beginning of each battle based on the training and maintenance preparation of the forces involved. These values then vary about these fixed values based on noise introduced by the confusion and stress of combat. We therefore postulate that for any given battle the random variables ($A(n)$, $Z_x(n)$, $B(n)$, $Z_y(n)$) can be represented by a constant plus some error (noise) random variable which is independent, identically distributed normal with mean zero and variance σ^2.

Thus we write:

$$\begin{aligned} -A(n) &= -a + \epsilon_A(n)\,, \\ Z_x(n) &= z_x + \epsilon_{Z_x}(n), \\ -B(n) &= -b + \epsilon_B(n)\,, \\ Z_y(n) &= z_y + \epsilon_{Z_y}(n). \end{aligned} \qquad (25)$$

Substituting these expressions into equation 24 yields the following system of equations:

$$X_{n+1} - X_n = [-a + \epsilon_A(n)]\, Y_n + [z_x + \epsilon_{Z_x}(n)],$$

$$Y_{n+1} - Y_n = [-b + \epsilon_B(n)]\, X_n + [z_y + \epsilon_{Z_y}(n)]. \quad (26)$$

It should be noted that in these equations the a, z_x, b, and z_y are only constant over the duration of battle. They will vary from battle to battle based on the combat preparation, force mix (e.g. the ratio of tanks to APCs), and the commander's concept of operations. Thus over the battle data set we will have a set of values $\left\{a_i, z_{x_i}, b_i, z_{y_i}\right\}$ for each battle which can be considered to be realizations of the random variables A, Z_x, B, and Z_y, where A, Z_x, B, and Z_y are jointly distributed with parameters to be estimated.

Finally, we include the individual attrition potential of each weapon system (fire power indices) from equation 22.

This completely specifies our model as follows:

$$X_{n+1} - X_n = [-a + \epsilon_A(n)]\, Y_n + [z_x + \epsilon_{Z_x}(n)],$$

$$Y_{n+1} - Y_n = [-b + \epsilon_B(n)]\, X_n + [z_y + \epsilon_{Z_y}(n)], \qquad (27)$$

where,

$$X_i = \sum_j f_{x_j} X_{ji}\,; \qquad Y_i = \sum_j f_{y_j} Y_{ji},$$

f_{x_j} ≡ the fire power index of X force weapon system type j,

f_{y_j} ≡ the fire power index of Y force weapon system type j,

X_{ji} ≡ the number of X weapon systems of type j active in the battle at the end of time increment i,

Y_{ji} ≡ the number of Y weapon systems of type j active in the battle at the end of time increment i,

$\epsilon_A(n)$ is iid Normal$(0, \sigma_A^{\,2})$,

$\epsilon_{Z_x}(n)$ is iid Normal$(0, \sigma_{Z_x}^{\,2})$,

$\epsilon_B(n)$ is iid Normal$(0, \sigma_B^{\,2})$,

$\epsilon_{Z_y}(n)$ is iid Normal$(0, \sigma_{Z_y}^{\,2})$,

(note $\epsilon_A(n)$, $\epsilon_{Z_x}(n)$, $\epsilon_B(n)$, *and* $\epsilon_{Z_y}(n)$ are also independent of each other)

$\left[a,\, z_x,\, b,\, z_y\right]$ *is a realization of* $\left[A,\, Z_x,\, B,\, Z_y\right]$

and a,b > 0.

E. FITTING TECHNIQUES

A number of techniques are evaluated in this thesis for estimating the parameters of a system of differential/difference equations from small battle trajectories. Using the non-linear programming solver MINOS 5.2 with the General Algebraic Modeling System (GAMS) to do a simultaneous least squares fit on the data was found to be the most effective. However, suppose one is willing to make the simplifying assumption that in an individual battle the coefficient random variables in one equation are independent of the coefficient random variable in the other equation (i.e that $[A, Z_x]$ and $[B, Z_y]$ are independent, but A and Z_x are still considered to be dependent, as are B and Z_y). In this case the parameters can be estimated by standard regression techniques. The validity of this assumption can be tested by fitting a small number of battle trajectories, using the GAMS model which considers each equation simultaneously, and comparing the resulting estimated coefficients with estimates obtained from the standard linear regression fits of the same battle trajectories (i.e. those obtained by considering each equation separately). If the comparison is favorable (i.e. the difference between the parameters fit by the two models is small), the savings in the computation time required for parameter estimation is great.

F. MEASURES OF BATTLE

In this section we consider measures derived from the difference equation model with only force size dependent terms; that is, all factors are considered to be dependent on force size. With this assumption equation 27 becomes:

$$X_{n+1} - X_n = [-a + \epsilon_A(n)]\, Y_n \ ,$$

$$Y_{n+1} - Y_n = [-b + \epsilon_B(n)]\, X_n \ . \qquad (28)$$

In defining the measures of battle the following definitions will be used.

ΔX_n ≡ the change in the fire power of the X force during the time increment [n, n+1]; so $\Delta X_n = X_{n+1} - X_n$.

ΔY_n ≡ the change in the fire power of the Y force during the time increment [n, n+1]; so $\Delta Y_n = Y_{n+1} - Y_n$.

X_n ≡ fire power of the X force at the beginning of the time increment [n, n+1].

Y_n ≡ fire power of the Y force at the beginning of the time increment [n, n+1].

N ≡ the number of time increments in the battle trajectory.

$\hat{a}$ ≡ estimate of a for a single battle; $$\hat{a} = \frac{\sum_{i=0}^{N-1} \frac{\Delta X_i}{Y_i}}{N}.$$

$\hat{b}$ ≡ estimate of b for a single battle; $$\hat{b} = \frac{\sum_{i=0}^{N-1} \frac{\Delta Y_i}{X_i}}{N}.$$

$\hat{a}_n$ ≡ the estimated value of a on [n, n+1]; $\hat{a}_n = \dfrac{\Delta X_n}{Y_n}$.

$\hat{b}_n$ ≡ the estimated value of b on [n, n+1]; $\hat{b}_n = \dfrac{\Delta Y_n}{X_n}$.

1. Pace of Battle

The pace of battle is defined to be: $\sqrt{\hat{a}\,\hat{b}}$. When our discrete model is fit to an entire battle trajectory the pace of battle provides an overall measure of the violence with which the battle was fought. It may be of even greater value when calculated at each individual time increment [n,n+1] of the battle thereby providing a picture of how the intensity of battle varied over time.

2. Relative Combat Power

The relative fire power is defined to be: $\dfrac{\hat{b}}{\hat{a}}$. When our model is fit to an entire battle trajectory, calculation of the relative combat power gives the average relative combat power of the two opposing forces. Calculation of the relative combat power during each time increment [n, n+1] again provides a picture of how relative combat power varies over time. This can be viewed as how the tide of battle fluctuates with time.

3. Probability of Victory

Based on our model (equations 28), the probability of victory for the X force can be approximated by: $P[X\ Wins] \approx P\left[\dfrac{Y_0^2}{X_0^2} < \dfrac{B}{A}\right]$. Note that this is only an approximation since we are ignoring the noise terms in equation 28. This measure allows us to measure how the distribution of the relative combat power and initial force ratios impact the battle outcome.

4. Elasticity

Our final measure, although not coming directly from our model, is composed of the same data elements as the previous measures. It is known as the elasticity [Ref. 9]:

$$\epsilon_n = \frac{\frac{\Delta X_n}{X_n}}{\frac{\Delta Y_n}{Y_n}} = \frac{\Delta X_n \, Y_n}{\Delta Y_n \, X_n} \, .$$

Calculating elasticity at each time interval, we obtain a picture of which force is winning or which force has the momentum. Elasticity is similar to relative combat power in that it also provides a picture of how the tide of battle varies over time.

5. Best Measures

Of these four measures, using a temporal trace of the pace of battle in conjunction with the relative combat power provides the most descriptive graphical representation of each battle process; providing a view of how the violence and tide of battle plays out over time. The probability of victory appears to provide us with a useful measure for analytical comparison.

Considering possible ways of computing the pace of battle and the relative combat power suggests two methods of comparison.

1. If we calculate each measure at each time interval then we get a temporal trace of the pace and relative combat power during the battle (see Figure 2). To compare these traces for two different battle trajectories we measure the distance between them (see Barr, Weir, Hoffman [Ref. 10]).

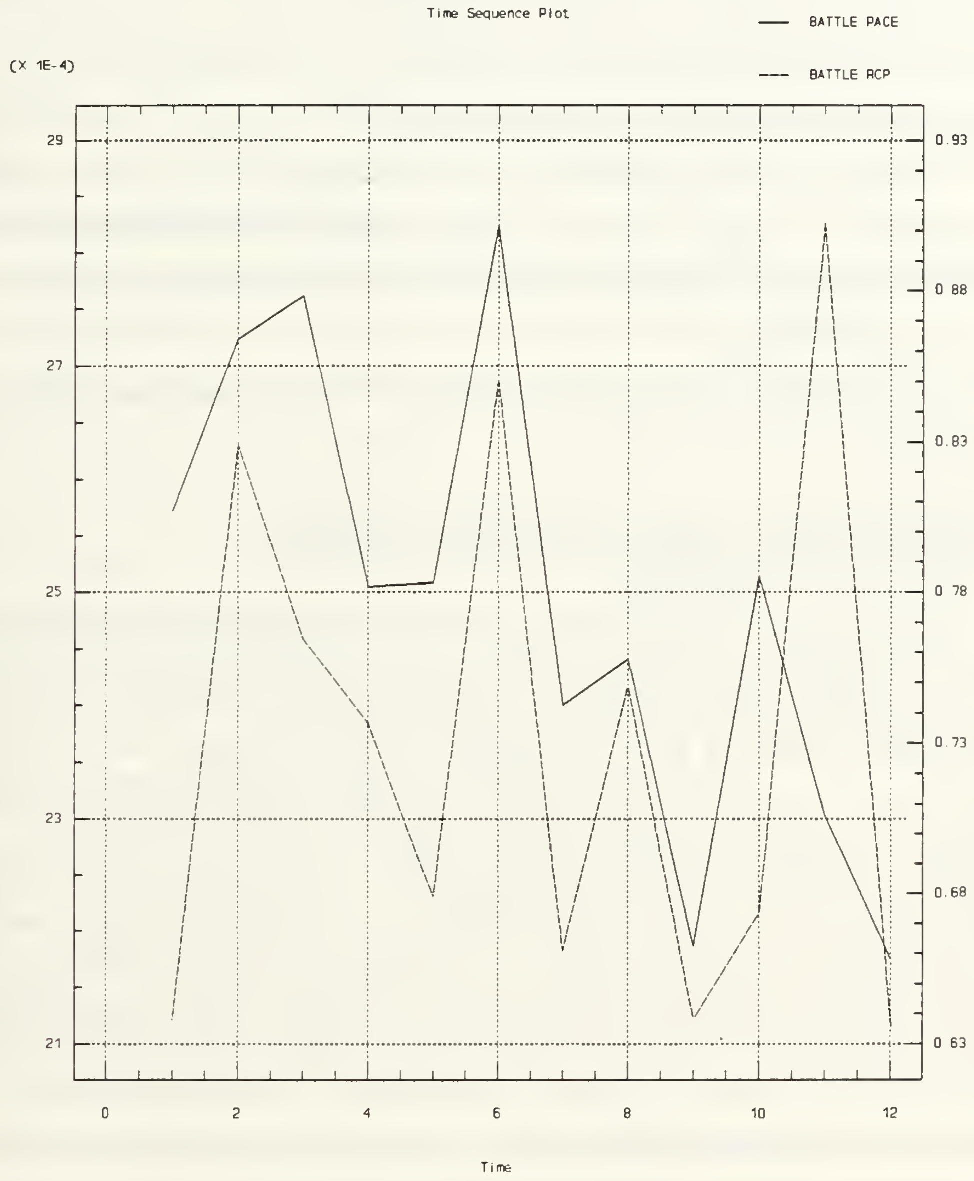

Figure 2 Temporal trace of pace and relative combat power.

2. If we calculate the average relative combat power

$$\hat{E}\left[\frac{B}{A}\right] = \frac{\sum_{i=0}^{N-1}\left(\frac{\hat{b}_i}{\hat{a}_i}\right)}{N},$$

where i ≡ index number of the battle trajectory,

N ≡ number of battle trajectories in the battle data set,

over each battle we might be able to compare these measures directly using the standard t-test since the distribution of relative combat power appears empirically to be normally distributed.

A methodology for using the probability of victory is described in detail in the next section.

G. COMPARING THE HIGH RESOLUTION MODELS

1. Methods of Comparison

Regardless of the fitting technique used for estimating the parameters of the model (see Chapter III), the process of fitting the proposed analytical models to the simulation data results in a set of estimates of the realizations of the coefficient random vectors. We desire to use the discrete model with the estimated coefficients to compare the high resolution simulation models. To do this, we need to identify measures that can be used to detect differences or similarities in the underlying structure of the high resolution models. Direct comparison of the coefficient random vectors can rely on independence between the coefficients and on sufficiently large data sets. To avoid having to make an undesirable independence assumption, and to utilize smaller, less costly data sets, we examined two indirect approaches for comparing our simulation models.

(1) Fighting simulated battles. This involves choosing coefficients from the empirical distributions of the fitted coefficients, and using them to run numerous

replications of a simulated battle. The MOEs from these simulated battles are then compared.

(2) Comparing Probability Surfaces. Using the estimated coefficient's empirical distribution we generate probability-of-win surfaces for various initial force ratios. Surfaces generated from two different battle data sets are compared to each other numerically to obtain a difference measure. A randomization test is then used to compare the two battle data sets.

2. Fighting Simulated Battles

Our first method uses the fitted analytical models to generate larger data samples for comparing the MOEs from each model being considered.

We draw pseudo coefficients from empirical distributions of the estimated coefficient random variable realizations. These coefficients are then substituted into the surrogate analytical models and a single battle is fought using this model, yielding a battle trajectory. We then draw a second set of coefficients and repeat the process. This continues for a predetermined number of replications.

From each simulated battle trajectory generated in this fashion we observe the MOE of interest. The empirical distributions of the MOEs corresponding to the models of the two simulation models are compared directly.

One difficulty with this method is determining the number of replications required to distinguish between two different battle data sets from two different simulation models while still being able to recognize when two battle data sets are from the same simulation model. We attack this problem by computing the sample means ($\hat{\mu}_1$ *and* $\hat{\mu}_2$), and sample variances ($\hat{\sigma}_1^2$ *and* $\hat{\sigma}_2^2$) of the MOE estimated directly with the data from the simulation model. This information is then used to estimate the number of observations required to obtain a confidence interval of length L on the sample means as follows:

$$\hat{n} = \left[\left(\frac{t_{1-\frac{\alpha}{2}}}{L} \right) \hat{\sigma}_{MAX}^{2} \right], where\ \hat{\sigma}_{MAX} = \max\{\hat{\sigma}_1, \hat{\sigma}_2\},$$ the maximum sample standard deviation of the MOE calculated with data from the two models (assuming normality for the distribution of the MOE), and $t_{1-\frac{\alpha}{2}}$ is the $1-\frac{\alpha}{2}$ quantile of a t distribution with degrees of freedom associated with $\hat{\sigma}^2_{\max}$.

This method of comparison tended to transform (or shift) the mean value of the MOE upward, and to introduce additional noise. That is, the variance of the MOE from the simulated (fitted) system was greater than the variance of the MOE in the original sample. Since our intent was to tighten the confidence interval on our estimated MOE, the introduction of additional noise was counterproductive, making the comparison less sensitive. Thus, although somewhat effective for identifying similarities and differences in the underlying battle structure, it was determined that this approach should not be pursued further.

3. Comparing Probability Surfaces

From our analysis in Section D, we know that if $(Y_0)^2/(X_0)^2 < b/a$ then the X force will be victorious. Since we are assuming that A and B are random variables, then R = B/A will also be a random variable. We can use observations of R obtained from our fitting process to build an empirical cumulative distribution function (cdf) $\hat{F}$ for R. This empirical cdf determines a probability surface F_R over the domain $\{(x_0,y_0) : x_l \le x_0 \le x_u\ ,\ y_l \le y_0 \le y_u\ \}$

where: $F_R(x_0,y_0) = P\left[R < \frac{x_0^2}{y_0^2} \right]$,

$x_0 \equiv$ the initial fire power of the X force,

$x_l \equiv$ the lower bound on the initial X fire power values,

$x_u \equiv$ the upper bound on the initial X fire power values, and similarly for the Y values.

Suppose we wish to compare two sets V and W of battle data (e.g. one consisting of several battle trajectories from Janus(A) and one consisting of several NTC battle trajectories). A natural measure for this comparison is the "volume" between the two surfaces:

$$\nu(V,W) = \sum_{y_0 = y_l}^{y_u} \sum_{x_0 = x_l}^{x_u} \left| F_{\Omega_V}(x_0,y_0) - F_{\Omega_W}(x_0,y_0) \right| \Delta x \, \Delta y ,$$

where Δx, Δy define the resolution of the partition of the domain of F_R ($\Delta x = \Delta y = 1$ was used in our trials).

A question is, "What does a given value of $\nu(V,W)$ mean?" For example, does a value of 800 indicate that the battle sets are the same or different? To address this question we first compare probability surfaces with a randomization test as follows.

Given our surrogate model (equation 28) and two battle data sets V and W consisting of n battle trajectories each: $V = \{v_1, v_2, \dots, v_n\}$ and $W = \{w_1, w_2, \dots, w_n\}$, where $v_i \equiv$ the i^{th} battle trajectory from simulation model V, and $w_i \equiv$ the i^{th} battle trajectory from simulation model W, we estimate the coefficients a and b for each battle trajectory in V and W. We denote these estimates by $\hat{a}_{v_i}, \hat{a}_{w_i}, \hat{b}_{v_i}, \textit{ and } \hat{b}_{w_i}$, where $\hat{a}_{v_i} \equiv$ the estimate of a obtained by fitting battle trajectory v_i, and the other estimates are defined in the same manner.

We define the sets R_v and R_w as follows:

$$R_v = \left\{r_{v_1}, r_{v_2}, \cdots, r_{v_n}\right\};\ \textit{where}\ r_{v_i} = \frac{\hat{a}_{v_i}}{\hat{b}_{v_i}},$$

and

$$R_w = \left\{r_{w_1}, r_{w_2}, \cdots, r_{w_n}\right\};\ \textit{where}\ r_{w_i} = \frac{\hat{a}_{w_i}}{\hat{b}_{v_i}}.$$

We now define an empirical cumulative distribution function $\hat{F}_R$ for a set of observed values of the random variable R; $\{r_{(1)}, r_{(2)}, \cdots, r_{(n)}\}$ where $r_{(1)} \le r_{(2)} \le \cdots \le r_{(n)}$.

$$\hat{F}_R(r) = \frac{\max\{i \mid r_{(i)} \le r\}}{n} \quad [\textit{Ref.}\ 13].$$

Examining the x_0-y_0 plane determined by a finite range of values for the initial fire power of the X force (X_0) and Y force (Y_0), that is, the plane $\{(x_0,y_0) \mid x_l \le x_0 \le x_u,\ y_l \le y_0 \le y_u\}$ where x_l, x_u, y_l, y_u are integer lower and upper bounds on the range of values for x_0 and y_0, we partitioned this plane into a grid of square intervals
$\{(x,y) \mid x_l+k \le x < x_l+(k+1),\ y_l+k \le y < y_l+(k+1)\}$, k = 0, 1, ... , m-1; where m = $x_u - x_l = y_u - y_l$.

We now use the definition of $\hat{F}_R$ and the partition on the domain of $\hat{F}_R$ to define a probability-of-win surface S. Given a set of observations of the random variable R, $\{r_1, r_2, \cdots, r_N\}$, of the ratios $\left(\frac{\hat{b}}{\hat{a}}\right)$, choose a random subset R^* of R.

We then define

$$S_{R^*} \equiv \left\{(x_0, y_0, f) \mid x_l \le x_0 < x_u,\ y_l \le y_0 < y_u,\ f = \hat{F}_{R^*}\left(\frac{y_0^2}{x_0^2}\right)\right\}.$$

We define our test statistic to be:

$$\nu\left(S_{R_1^*}, S_{R_2^*}\right) = \sum_{y_0=y_l}^{Y_u} \sum_{x_0=x_l}^{x_u} \left| \hat{F}_{R_1^*}\left(\frac{y_0^2}{x_0^2}\right) - \hat{F}_{R_2^*}\left(\frac{y_0^2}{x_0^2}\right) \right|.$$

$\nu(S_{R_1^*}, S_{R_2^*})$ is the volume between the probability-of-win surfaces generated by the sets R_1^* *and* R_2^*, where R_1^* *and* R_2^* are sets of observed values of the random variable R.

To determine if the two simulation models V and W are different, we first compute $T = \nu(S_{R_V}, S_{R_W})$. This is the observed value of our test statistic. We then form $R = R_V \cup R_W$. Let P be the set of all possible partitions R into two subsets R_1 and R_2 such that both R_1 and R_2 contain n elements and $R_1 \cap R_2 = \emptyset$. A specific partition p ϵ P can be built by selecting n elements of R at random (without replacement) as the set R_1 and letting the set R_2 be $R \backslash R_1$. We can then compute $\nu(S_{R_1}, S_{R_2})$. Our randomization distribution is defined to be the set of values $\nu(S_{R_1}, S_{R_2})$ for all p ϵ P. We compare our observed value T with a sample from the randomization distribution, obtained by randomly choosing M partitions p ϵ P and evaluating $\nu(S_{R_1}, S_{R_2})$ for each p. If the value of T is in the right tail of this sampled distribution (i.e. an extreme value) we infer that V and W are from different simulation models. And if T is in the left tail of the randomization distribution we infer that V and W are from the same simulation model.

Since the number of values in the randomization distribution may be as high as $\frac{1}{2}\binom{2n}{n} = \frac{(2n)!}{4(n!)}$, it would be impractical to compute each value. However, it is reasonable to take a sample from this distribution for testing the observed value of T [Ref. 1].

In our studies, this technique proved to be effective (results are given in Chapter IV) at identifying difference in battle data sets V and W, when V and W came from different generating systems (see Chapter IV , Section A for a description of the generating systems used), while still recognizing similarity, when V and W were generated by the same system. This technique is demonstrated in Chapter IV.

III. FITTING TECHNIQUES

A. INTRODUCTION

Our approach to finding an appropriate technique for fitting the coefficients of a system of simultaneous difference equations has been experimental. We first examined the simulated annealing technique suggested by Ingber [Ref. 2]. We then considered methods that required less computation time. This chapter is a chronological account of our research, we thus present our results in that order.

B. SIMULATED ANNEALING

The simulated annealing algorithm was developed by Metropolis in 1953 [Ref. 11] to simulate the physical annealing process studied in statistical mechanics. It was first suggested as a technique for solving combinatoric optimization problems in 1983 by Kirkpatrick, Gelatt, and Vecchi [Ref. 12]. The simulated annealing algorithm combines local optimization techniques with a random walk process to reduce the chance of becoming trapped in a local optimum. The algorithm shown in Figure 3 begins with an initial solution and generates a proposed neighboring solution at random. If the proposed solution is better than the current solution (i.e. a *downhill move*) then the proposed solution is accepted as the new current solution with probability one. If the proposed solution is worse than the current solution (i.e. a *uphill move)* then it is accepted with a probability based on the magnitude of the uphill move and the current value of the control parameter (referred to as the temperature of the process). Thus, small uphill moves are more likely to be accepted than large ones, and all uphill moves are more likely to be accepted at higher temperatures than at lower temperatures. The process is run for a large number of repetitions at each temperature, and reductions in temperature are made according to some "cooling schedule." Changes to the initial solution, initial temperature, and cooling schedule can have dramatic effects on the convergence properties of the algorithm and most often must be arrived at by experimentation.

Simulated annealing's main utility is in solving hard combinatorial optimization problems for which:

- no good problem specific heuristic algorithms have been developed (simulated annealing does not compete well with problem specific techniques, like those developed for the traveling salesman problem, see Johnson, Aragon, McGeoch, and Schevon (1989) [Ref. 14]);

```
1. Get an initial solution S.
2. Get an initial temperature T>0.
3. While not yet frozen do the following.
   3.1 Perform the following loop L time.
       3.1.1 Pick a random neighbor S' of S.
       3.1.2 Let $\Delta$ = cost(S') - cost(S).
       3.1.3 If $\Delta \leq 0$ (downhill move),
               Set S = S'.
       3.1.4 If $\Delta > 0$ (uphill move),
               Set S = S' with probability $e^{-\Delta/T}$.
   3.2 Set T = rT (reduce temperature).
4. Return S.
```

Source: JOHNSON ET AL. [Ref 13].

Figure 3. Generic Simulated Annealing Algorithm.

- there is limited application for the formulated problem (if the problem formulation has wide applicability efforts toward developing an effective problem specific algorithm should be more beneficial);
- the solution space is well understood (since the annealing algorithm is highly parameter dependent, a strong understanding of the solution space is critical for ensuring convergence of the algorithm to a global optimum).

In a proposal to the U.S. Army TRADOC Analysis Command [Ref. 3], Ingber proposed that a variation of the conventional simulated annealing algorithm, referred to as Very Fast Simulated Reannealing, could be used effectively to fit systems of differential equations to battle data from various high resolution simulations and NTC. Our initial experiments with simulated annealing were designed to use a somewhat simplified version of Ingber's algorithm to assess his claim. We first attempted to fit individual battle trajectories and various subsets of battle data sets to a system of differential equation of the following form (our difference equation model was developed later):

$$\frac{d\,RTANK}{dt} = a_{RTANK,BTANK} BTANK + a_{RTANK,BAPC} BAPC + a_{RTANK,BTOW} BTOW$$

$$\frac{d\,RAPC}{dt} = a_{RAPC,BTANK}\ BTANK + a_{RAPC,BAPC}\ BAPC + a_{RAPC,BTOW}\ BTOW$$

$$\frac{d\,BTANK}{dt} = a_{BTANK,RTANK} RTANK + a_{BTANK,RAPC} RAPC$$

$$\frac{d\,BAPC}{dt} = a_{BAPC,RTANK}\ RTANK + a_{BAPC,RAPC}\ RAPC$$

$$\frac{d\,BTOW}{dt} = a_{BTOW,RTANK}\ RTANK + a_{BTOW,RAPC}\ RAPC$$

where RTANK denotes the number of red tanks involved in the battle at time t, $a_{RTANK,BTANK}$ is the attrition rate of red tanks by blue tanks, and similarly for the other variables and coefficients (note here that APC denotes an armored personnel carrier, and TOW denotes the Tube-Launched, Optical-tracked, Wire-guided antitank weapon system of the U.S. Army).

After numerous experimental runs of the simulated annealing algorithm it became very clear that regardless of the parameters used in the algorithm, when a small number of battles were used, the results were quite unstable. In particular, starting with one seed for the pseudo-random number generator we would get one solution and starting with a different seed we would arrive at a different solution using the same data. When a larger number of battles were used the computer time required became unreasonable (on the order of 24 hours on a 386-based PC running at 33 Mhz when fitting ten battle trajectories). Based on these results we temporarily abandoned simulated annealing and searched for quicker and more stable techniques. We discuss several such techniques in the remainder of this chapter. We found during our experiments with various fitting methods that it was reasonable to specify relative fire power indices for each weapon system, thus allowing us to fit a simpler system of differential equations given below:

$$\frac{dX}{dt} = -aY + Z_x,$$

$$\frac{dY}{dt} = -bX + Z_y,$$

$$where,$$

$$X = f_{RTANK}RTANK + f_{RAPC}RAPC,$$

$$Y = f_{BTANK}BTANK + f_{BAPC}BAPC + f_{BTOW}BTOW.$$

Experimental runs using simulated annealing for fitting this system of equations were both stable and much quicker than those experienced previously. However, we observed that the simulated annealing process simply converged to the regression fits discussed in Section E, and took about 10000 times longer to do so. Thus, simulated annealing was found to be an unsuitable technique for fitting systems of equations of the form in which we were interested.

C. STEEPEST DESCENT

While trying to determine why simulated annealing was not working well, we made two observations regarding our model. First, Lanchester models assume that every weapon system on the battlefield can see and engage every other system on the battlefield. This was not true of either the battles observed at NTC or those generated by Janus(A). Second, since the models under consideration were attrition models, their solution spaces were convex, and thus a least squares measure (i.e. the sum of the square differences) of the fit of the data to these models would also induce a convex space. Thus a simpler fitting technique such as a standard steepest descent algorithm should be effective.

At this point we consulted with analysts at the TRADOC Analysis Command (TRAC) and determined that although battle data showing which systems could see and engage which other systems was not currently available, the Janus(A) system could be modified to produce such data. Thus, we proceeded by generating simulated battle trajectories from mathematical models to use in testing our fitting techniques (see Chapter IV, Section A).

We began testing the straight-forward steepest descent algorithm shown in Figure 4, using this simulated data (FORTRAN code is in Appendix C).

This algorithm proved to be stable with respect to the quality of solution, but not with respect to time of solution. That is, regardless of the starting solution , the algorithm would converge to the same solution for a given accuracy level; however, the time required to do so varied from one minute up to four hours. Although this algorithm was far more useful than the simulated annealing approach, the idea of using non-linear fitting techniques led us to try using commercial non-linear programming software to fit our system of equations.

D. NON-LINEAR OPTIMIZATION

The convex nature of our problem encouraged us to believe that we might be able to use commercial non-linear solvers to fit our equations to the battle data. We thus formulated our problem on the General Algebraic Modeling System (GAMS) using the MINOS 5.1 solver (see code in Appendix D). We made experimental trials using both a least squares measure (i.e Euclidean distance) and a Chebychev (or minimax) criterion.

Least Squares Objective Function: *Min* $\sum_i \left(\delta_{x_i}^{\,2} + \delta_{y_i}^{\,2}\right)$

Chebychev Objective Function: *Min* $\left(\max\left\{|\delta_{x_i}| + |\delta_{y_i}|\right\}\right)$

Subject to: $\delta_{x_i} \geq (x_{i+1} - x_i) - (a\, y_i + z_x)$

$$\delta_{y_i} \geq (y_{i+1} - y_i) - (b\, x_i + z_y)$$

Solve for: $\hat{a}_{OPT},\ \hat{b}_{OPT},\ (\hat{z}_x)_{OPT},\ (\hat{z}_y)_{OPT}$

```
1. Get an initial solution S.
2. Determine an initial grid definition size.
3. While not yet close enough:
    3.1 Compute cost for each neighboring point on the grid.
    3.2 Set S'= neighboring solution with min cost.
    3.3. IF $\|S - S'\|$ < accuracy THEN refine the grid size.
4. Return S = S'.
```

Figure 4. Steepest Descent Algorithm.

where $\delta_{x_i} \equiv$ difference between the predicted attrition and the actual attrition.

Our results were encouraging. The solutions were stable and were found consistently in just under one minute for single battles. We also observed graphically that the least squares criterion provided a smaller variance on the fitted coefficients.

E. LINEAR REGRESSION

Our GAMS formulation with a least squares objective function was essentially providing a simultaneous linear regression fit of the battle trajectories to the equations in the analytical model being fit. This led us to ask when this would be the equivalent or nearly equivalent to doing linear regression on each of the individual equations separately. The answer is that if we assume the coefficients in the equations are independent, then we could simply fit the equations separately using regression.

Considering the method (Chapter IV, Section A) used to generate our simulated battle trajectories (which held the attrition and noise coefficients constant for the duration of each individual battle), this independence relationship was true for the individual battle trajectories (this would not be the case for the data in general). We did regression fits on the individual equations and found the two methods equivalent (i.e. the fitted parameters were nearly equal) up to a factor that could be explained by noise induced by the generation of integer data points.

Thus, if we are willing to make the assumption that the equations are independent for individual battles, we reduce our fitting time from about one minute per battle to about 0.1 seconds per battle. Whether this makes sense to do on actual battle trajectories needs to be examined once the data become available.

F. CONCLUSION

Our experiments with the non-linear optimization approach of using a GAMS model with a least squares objective function indicate that this approach is the most effective way of fitting a system of difference equations to battle trajectories, providing consistent solutions in a timely manner. However, experiments should be made to consider whether assuming independence between the coefficients in each equation of the surrogate model is justified, thus allowing us to use the faster independent regression technique discussed in the last section. We use the independent regression technique for an example in the next chapter.

IV. EXAMPLE OF METHOD

A. THE BATTLE DATA TRAJECTORIES

To test the methodology, we first generated battle trajectory data based on a discrete version of Lanchester's aimed fire model with additive noise term (equation 29 below). Three different systems of equations (all of this same form) were used for this purpose. These generating systems differed only in the fire power indices assigned to each weapon system. Each system was used to generate two separate collections of 20 battles each. The form of the generating equations is:

$$X_{n+1} - X_n = -AY_n + Z_x,$$

$$Y_{n+1} - Y_n = -BX_n + Z_y, \qquad (29)$$

where

$$X_n = f_{RTANK} RTANK_n + f_{RAPC} RAPC_n,$$

$$Y_n = f_{BTANK} BTANK_n + f_{BAPC} BAPC_n + f_{BTOW} BTOW_n .$$

and A and B are constant throughout a single battle trajectory. Here, f_{RTANK} is the fire power index for the red tanks, and similarly for the fire power indices of the remaining weapon systems (values shown in Table 2 below). $RTANK_n$ is the number of red tank systems active in the battle at the end of time increment n. $RTANK_n$ is computed as follows from X_n. The other weapon system levels (RAPC, BTANK, BAPC, and BTOW) are all computed in the a similar manner. At the end of each time interval [n, n+1], the attrition sustained by the X force is divided among the two X weapon systems as follows:

$$RTANK_{n+1} = RTANK_n + (X_{n+1} - X_n)\left[\frac{tfpwr_X - f_{RTANK}}{(tfpwr_X)(Xsys - 1)}\right],$$

$$RAPC_{n+1} = RAPC_n + (X_{n+1} - X_n)\left[\frac{tfpwr_X - f_{RAPC}}{(tfpwr_X)\ (Xsys - 1)}\right],$$

where

$$tfpwr_X = f_{RTANK} + f_{RAPC},$$

Xsys ≡ the number of different weapon system in X force,

and similarly for the Y force. We then round $RTANK_{n+1}$ and $RAPC_{n+1}$ down to the next integer value.

Also, $[A\ Z_x\ B\ Z_y]$ is distributed multivariate normal with variance-covariance matrix:

$$V = \begin{bmatrix} 0.00004 & -0.00200 & -0.00001 & 0.00010 \\ -0.00200 & 0.50000 & -0.00060 & 0.40000 \\ -0.00001 & -0.00060 & 0.00008 & -0.00600 \\ 0.00010 & 0.40000 & -0.00600 & 1.00000 \end{bmatrix},$$

and mean $\mu = [\ 0.066\ 0.000\ 0.055\ 0.000\ ]$.

Initial force levels were

RTANK(0) = 80
RAPC(0) = 160

BTANK(0) = 60
BAPC(0) = 120
BTOW(0) = 40 ,

The fire power indices f were set as shown in Table 2:

TABLE 2. FIRE POWER INDICES.

BATTLE SET	A & B	C & D	E & F
RTANK	0.90	1.00	0.90
RAPC	0.80	0.90	0.80
BTANK	1.00	0.90	0.90
BAPC	0.90	0.80	0.90
BTOW	0.80	0.70	0.25

A realization of [A, Z_x, B, Z_y] was chosen and a battle trajectory consisting of 12, five minute intervals was run out using the generating equations (29). Another realization was then chosen and another battle trajectory was generated. In this manner 20 battle trajectories were generated for each battle data set. A sample battle trajectory generated with these equations is shown in Table 3.

TABLE 3. BATTLE TRAJECTORY SAMPLE.

Battle Trajectory (Battle 1 Set A)					
TIME	RTANK	RAPC	BTANK	BAPC	BTOW
0	80	160	60	120	40
5	73	152	57	117	37
10	67	145	54	114	33
15	61	138	51	111	30
20	55	132	49	108	27
25	49	126	46	106	25
30	44	120	44	103	22
35	39	114	42	101	20
40	34	109	40	99	18
45	30	103	39	97	16
50	25	98	37	96	14
55	21	94	35	94	13
60	17	89	34	93	11

B. SELECT MATHEMATICAL SURROGATE SYSTEM

We had decided to use the Lanchester aimed fire model in our initial tests. However, we had to decide whether to include the additive (random) noise term. The concern was to choose the model that would best represent the underlying structure of the data represented by the generating system. The best candidate was determined by running a small sample test. Five battles were selected at random from battle set A (see Table 3) and were fit to the discrete form of the Lanchester aimed fire model with and without the constant noise terms. The optimization model (procedure) used for estimating the parameters in the following surrogate model without additive noise terms

$$X_{n+1} - X_n = \left[-a + \epsilon_A(n)\right]Y_n\,,$$
$$Y_{n+1} - Y_n = \left[-b + \epsilon_B(n)\right]X_n\,, \qquad (30)$$

is given below.

Proportional Procedure (no additive noise term):

$$\textit{Minimize:} \quad \sum_i \left(\delta_{x_i}^{\;2} + \delta_{y_i}^{\;2}\right)$$

$$\textit{subject to:} \quad \delta_{x_i} \geq (X_{i+1} - X_i) - (a\,Y_i)$$

$$\delta_{y_i} \geq (Y_{i+1} - Y_i) - (b\,X_i)$$

$$\textit{Solve for:} \quad \hat{a}_{OPT},\ \hat{b}_{OPT}$$

where δ_{X_i} *and* δ_{Y_i} are as in Chapter II.

Additionally, the surrogate model with the additive noise term (equation 27) was fit using a one-stage and two-stage fitting procedure. The one-stage procedure was the non-linear programming technique describe in Chapter III.

One-stage Procedure:

$$\textit{Minimize:} \quad \sum_i \left(\delta_{x_i}^{\;2} + \delta_{y_i}^{\;2}\right)$$

$$\textit{subject to:} \quad \delta_{x_i} \geq (X_{i+1} - X_i) - (a\,Y_i + z_x)$$

$$\delta_{y_i} \geq (Y_{i+1} - Y_i) - (b\,X_i + z_y)$$

$$\textit{Solve for:} \quad \hat{a}_{OPT},\ \hat{b}_{OPT},\ \hat{z}_{x\,OPT},\ \hat{z}_{y\,OPT}$$

The two-stage procedure used the same basic technique; however, in stage one the noise terms were fixed at zero and the attrition coefficients were fit (this stage is the same as the proportional procedure). In the second stage the attrition coefficients were fixed at the optimal value found in stage one and the noise term was fit.

Two-stage procedure:

Stage One Problem:

$$\text{Minimize:} \quad \sum_i \left(\delta_{x_i}^{\,2} + \delta_{y_i}^{\,2}\right)$$

$$\text{subject to:} \quad \delta_{x_i} \geq (X_{i+1} - X_i) - (a\,Y_i)$$

$$\delta_{y_i} \geq (Y_{i+1} - Y_i) - (b\,X_i)$$

$$\text{Solve for:} \quad \hat{a}_{OPT}, \hat{b}_{OPT}$$

Stage Two Problem:

$$\text{Minimize:} \quad \sum_i \left(\delta_{x_i}^{\,2} + \delta_{y_i}^{\,2}\right)$$

$$\text{subject to:} \quad \delta_{x_i} \geq (X_{i+1} - X_i) - (\hat{a}_{OPT} Y_i + z_x)$$

$$\delta_{y_i} \geq (Y_{i+1} - Y_i) - (\hat{b}_{OPT} X_i + z_y)$$

$$\text{Solve for:} \quad \hat{z}_{x\,OPT}, \hat{z}_{y\,OPT}$$

The results are shown in Tables 4, 5, and 6, where cost is taken to be the square root of sum of square differences of the predicted and observed attrition from the model (equation 29).

TABLE 4. ONE-STAGE FITTING.

Results of One-Stage Fitting Procedure					
BATTLE	A-1	A-2	A-3	A-4	A-5
COST	4.522	6.056	3.544	4.724	7.030
$\hat{a}$	0.072	0.054	0.056	0.054	0.081
$\hat{z}_x$	1.631	0.799	1.290	-0.052	2.138
$\hat{b}$	0.046	0.058	0.045	0.056	0.062
$\hat{z}_y$	0.650	1.545	1.191	1.029	1.863

TABLE 5. TWO-STAGE FITTING.

Results of Two-Stage Fitting Procedure					
BATTLE	A-1	A-2	A-3	A-4	A-5
COST	4.675	8.742	3.720	5.700	7.938
$\hat{a}$	0.061	0.049	0.048	0.054	0.067
$\hat{z}_x$	0.030	0.024	0.021	-0.002	0.056
$\hat{b}$	0.042	0.048	0.037	0.049	0.049
$\hat{z}_y$	-3.519	0.176	-1.764	-0.539	3.082

TABLE 6. PROPORTIONAL FITTING.

Results of Proportional Fitting Procedure					
BATTLE	A-1	A-2	A-3	A-4	A-5
COST	5.378	7.047	4.352	5.147	10.991
$\hat{a}$	0.061	0.049	0.048	0.054	0.056
$\hat{b}$	0.042	0.048	0.037	0.049	0.049

Examining the costs (sum of squares error) in the tables above we concluded that the one-stage fitting procedures was the best (minimum error) of the three examined.

We were however interested in obtaining a fitted model that best represents the underlying structure of the data that we were fitting. Thus, we also looked at the Euclidean-norm of the difference between the fitted coefficients and those used to generate the data. Let $\alpha = [A, Z_x, B, Z_y]$; define $\hat{\alpha}$ to be the parameters estimated by the fitting procedure (take $Z_x = Z_y = 0$ in the proportional procedure). Let $\alpha_g = \left[\mu_A, \mu_{Z_x}, \mu_B, \mu_{Z_y}\right]$, the mean values of the parameters used in the generating equations to simulate the battle data sets. We then compute $\| \hat{\alpha} - \alpha_g \| = \sqrt{(\hat{a} - \mu_A)^2 + (\hat{z}_x - \mu_{Z_x})^2 + (\hat{b} - \mu_B)^2 + (\hat{z}_y - \mu_{Z_y})}$. These values are shown in Table 7.

TABLE 7. NORM OF DIFFERENCES IN COEFFICIENT VECTORS.

Euclidean-norm of difference in Coefficients						
BATTLE	A-1	A-2	A-3	A-4	A-5	AVG
2-Stage	2.59	0.04	1.96	1.78	3.10	1.85
1-Stage	2.24	1.88	1.56	2.22	2.41	2.06
Pro.	0.93	0.25	0.21	1.45	0.58	0.63

We also examined the battle trajectories for each of these five fitted surrogate models graphically, by plotting the force levels versus time for the fitted system and the generating system (see Figures 5-7 for an example).

As a result of the analyses we concluded that although the one-step fit was providing the best fit in the least squares sense; when we examined the results using the proportional fit on each individual equation (in equation 28) independently (Figure 8) the fit to the underlying structure (i.e. the generating equations) was providing a graphical representation almost as good as the one-stage procedure. Since the proportional procedure used was much quicker than the one-stage procedure we decided to use it. The following optimization model is the procedure

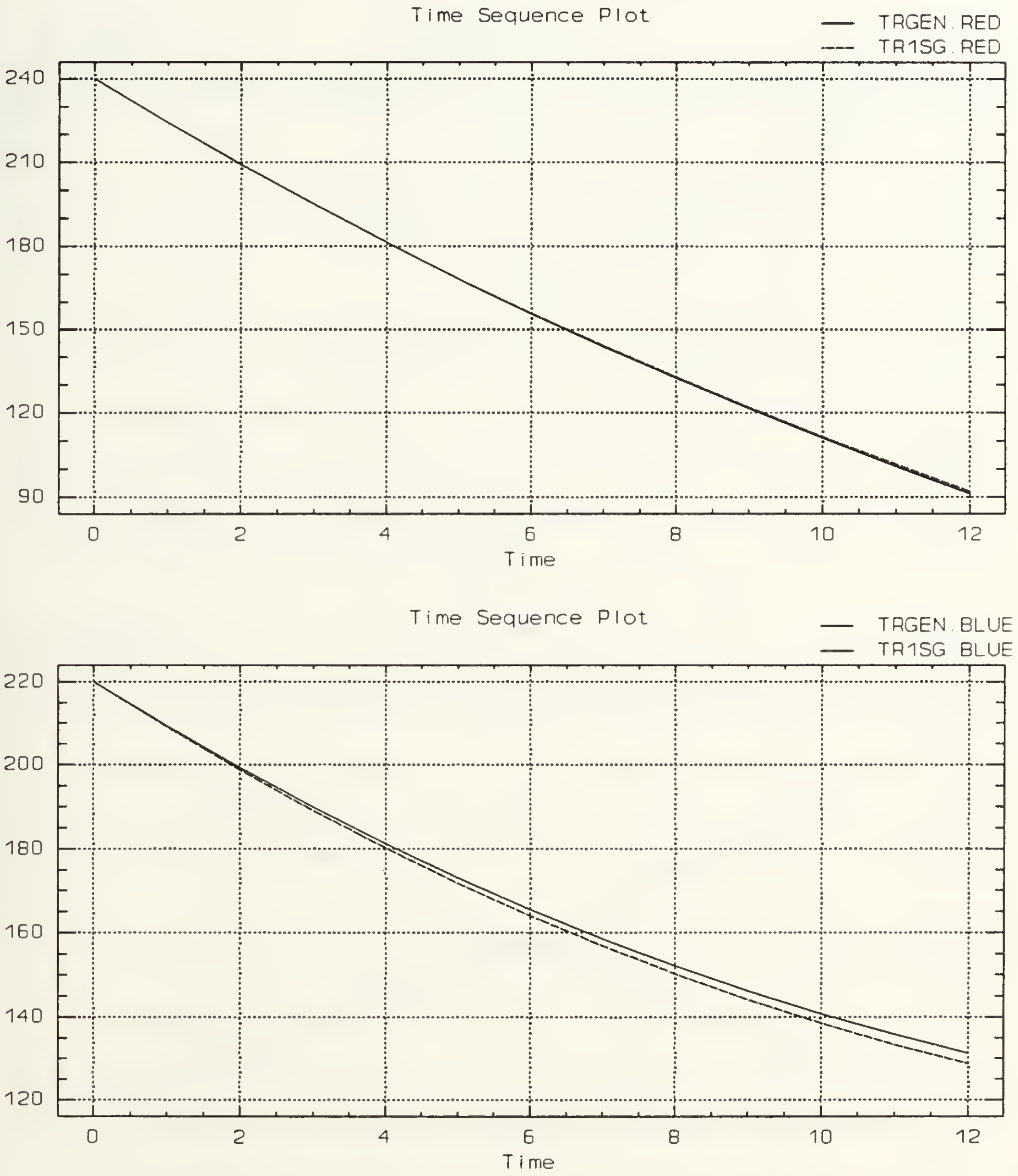

Figure 5. Battle Trajectory Plots. (1-Stage vs Gen)

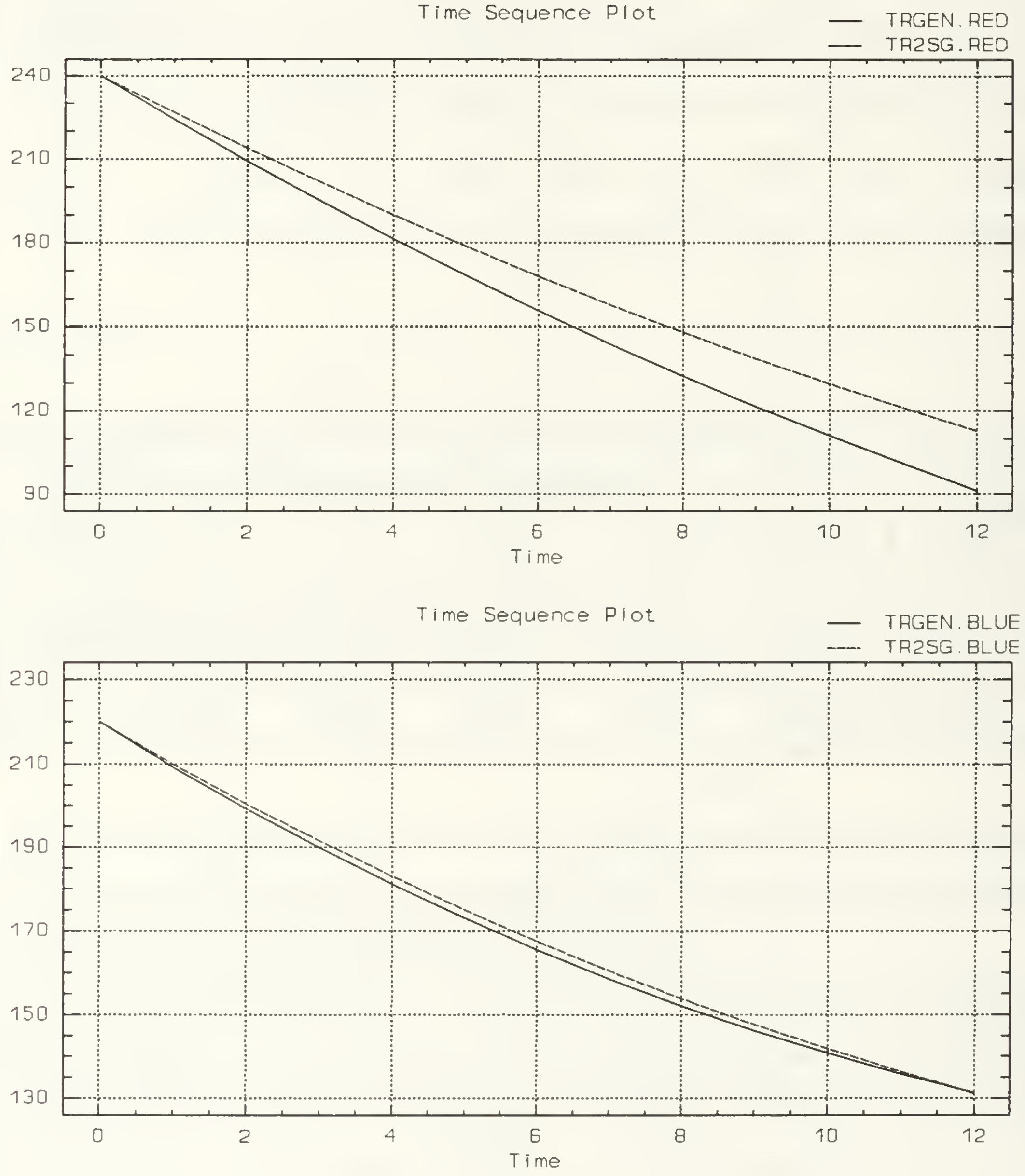

Figure 6. Battle Trajectory Plots. (2-Stage vs. Gen)

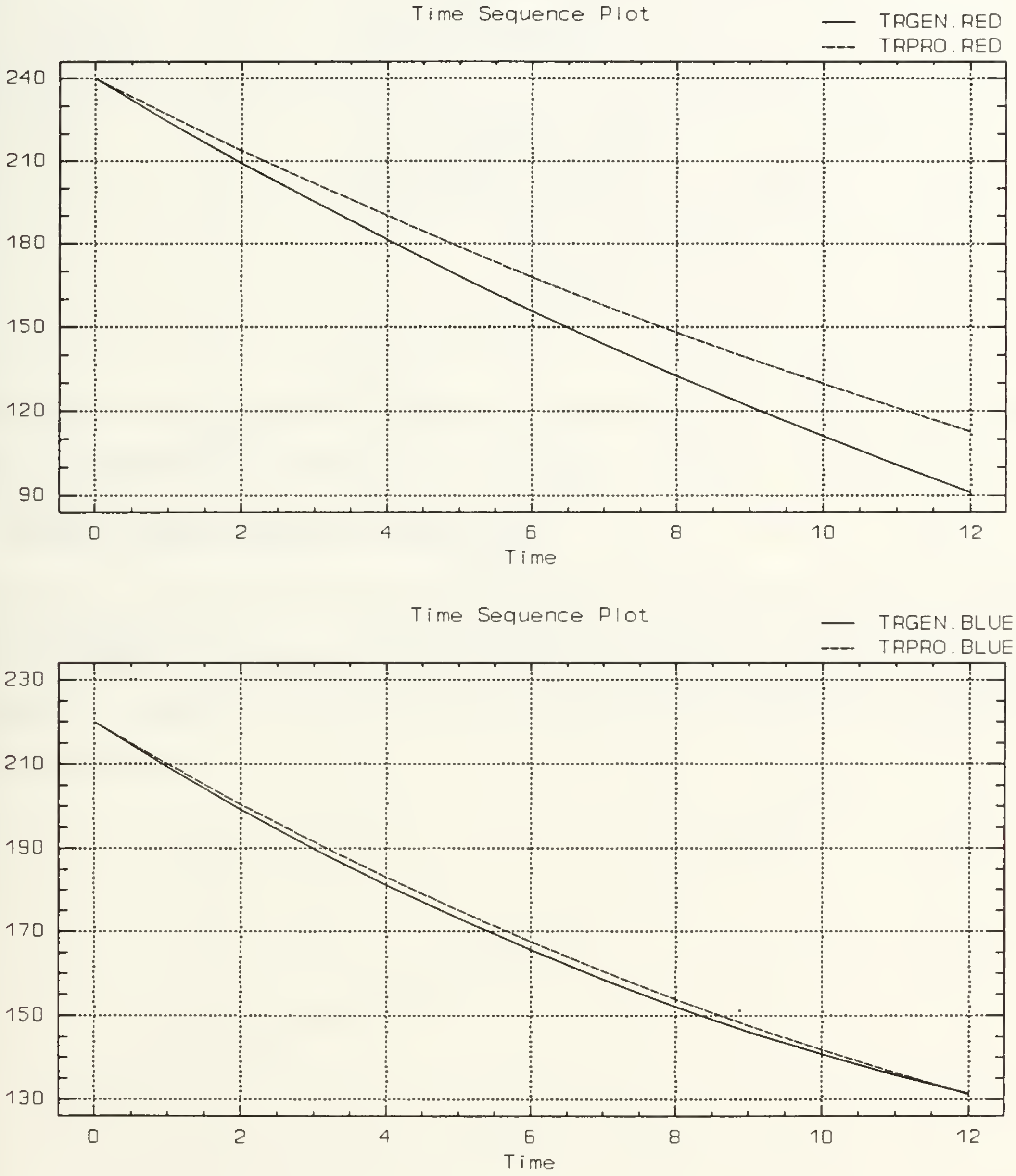

Figure 7. Battle Trajectory Plots. (Pro vs. Gen)

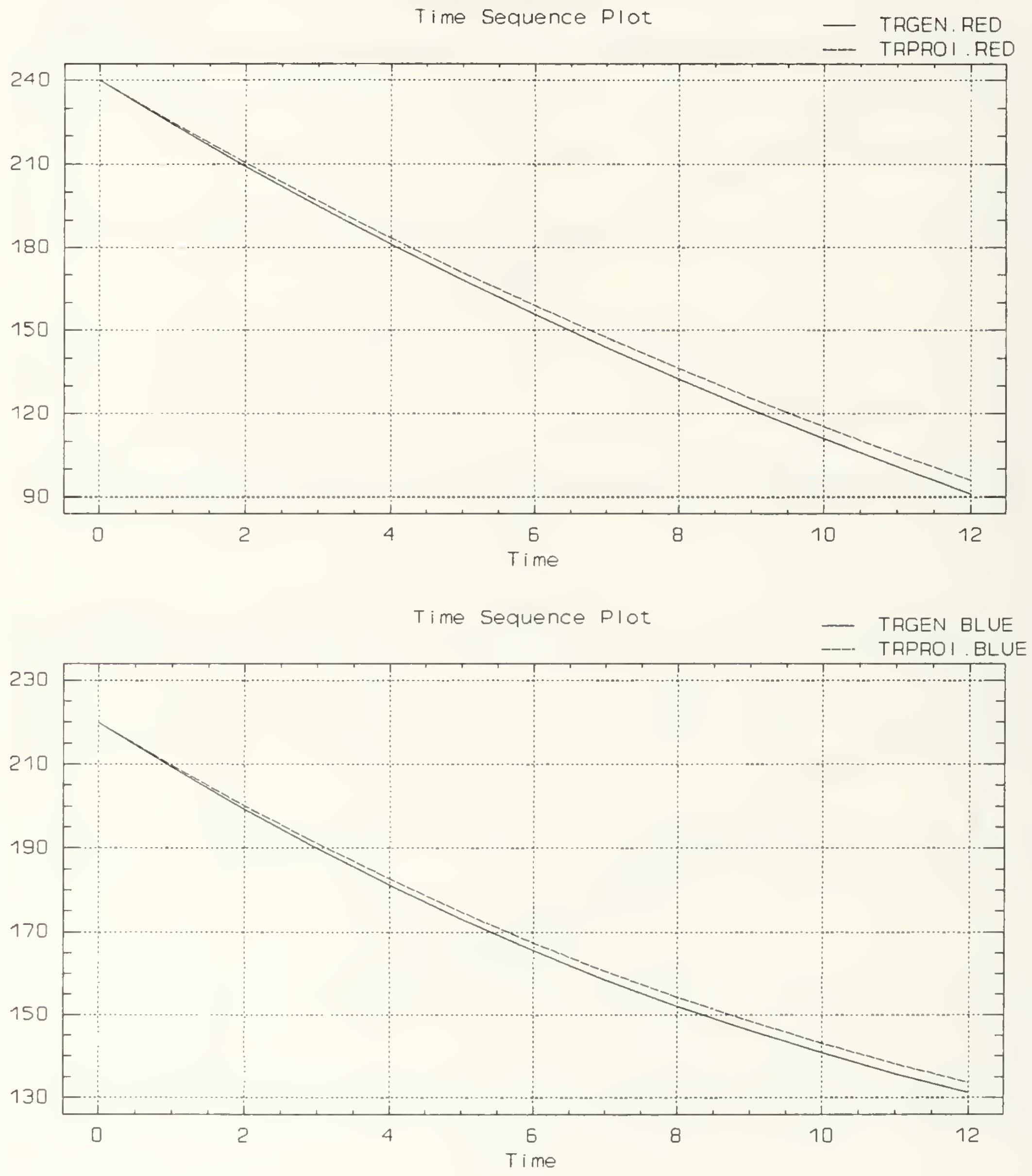

Figure 8. Battle Trajectory Plots. (Pro-I vs. Gen)

used.

Proportional independent procedure (PRO-I):

$$\hat{a} = \frac{\sum_{i=0}^{N-1}\left(\frac{X_{i+1} - X_i}{Y_i}\right)}{N}$$

$$\hat{b} = \frac{\sum_{i=0}^{N-1}\left(\frac{Y_{i+1} - Y_i}{X_i}\right)}{N}$$

We therefore used the discrete difference equation model without additive noise (equation 30) for the remainder of the analysis.

C. FITTING THE MODEL TO THE DATA

Based on the analysis in the preceding section, the model was fit to the battle data sets obtained from the generating systems using the proportional independent method. The coefficients fit to the first five replications of each data set are shown in Table 8 below.

TABLE 8. FITTED COEFFICIENTS.

Sample of Fitted Coefficients						
Battle	Set A	Set B	Set C	Set D	Set E	Set F
1: $\hat{a}$ $\hat{b}$	0.0586 0.0426	0.0502 0.0505	0.0493 0.0499	0.0406 0.0582	0.0499 0.0417	0.0414 0.0466
2: $\hat{a}$ $\hat{b}$	0.0469 0.0487	0.0489 0.0530	0.0383 0.0548	0.0379 0.0592	0.0387 0.0451	0.0394 0.0474
3: $\hat{a}$ $\hat{b}$	0.0520 0.0400	0.0533 0.0396	0.0409 0.0515	0.0447 0.0459	0.0428 0.0411	0.0449 0.0391
4: $\hat{a}$ $\hat{b}$	0.0509 0.0524	0.0573 0.0499	0.0422 0.0580	0.0469 0.0572	0.0423 0.0469	0.0479 0.0464
5: $\hat{a}$ $\hat{b}$	0.0600 0.0503	0.0572 0.0470	0.0505 0.0584	0.0463 0.0551	0.0512 0.0467	0.0476 0.0446
Mean: $\hat{a}$ $\hat{b}$	0.0507 0.0474	0.0570 0.0480	0.0454 0.0548	0.0471 0.0555	0.0462 0.0448	0.0479 0.0452

D. EMPIRICAL DISTRIBUTION FUNCTIONS

Graphical analysis of the fitted coefficients from the battle data sets (of 20 battle trajectories each) show no clear fit to a known probability distribution, thus we turned to the empirical distributions. The empirical cumulative distribution functions for the ratio of the estimated coefficients $\frac{\hat{b}}{\hat{a}}$ where determined for each battle (see Figure 9 below).

Using the procedure outlined in Chapter II we generated the probability surfaces numerically over square intervals and calculated the difference between surface heights at the corner of each square interval $\{X \leq x_0 < X+1, Y \leq y_0 < Y+1\}$. The observed values of our test statistic T (volume between surfaces compared) of the

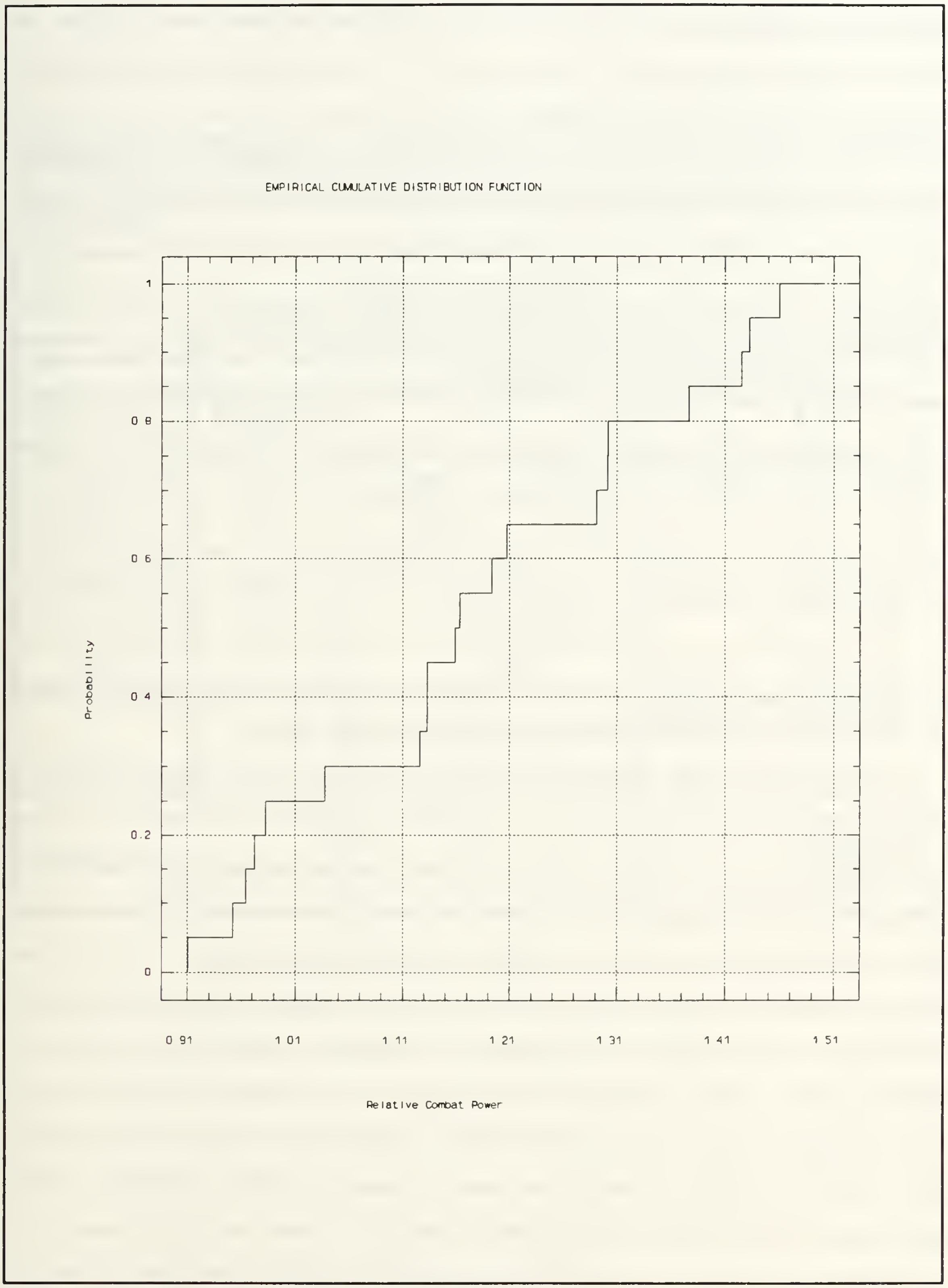

Figure 9. Empirical CDF for B/A (Battle Data Set A).

battle set comparisons are listed in Table 9. The notched box plots in Figure 10 provide a graphical portrayal of samples of size 50 of the randomization distribution of these differences and the location of the observed test statistic within this distribution. Examination of these box plots clearly delineates the differences between the dissimilar battle data sets, and the similarity of the equivalent battle data sets (i.e. A and B, C and D, and E and F as defined in Table 3).

TABLE 9. OBSERVED VALUE OF THE TEST STATISTIC T.

Observed Value of Test Statistic					
vs	B	C	D	E	F
A	201	2940	2849	1079	928
B		3023	2932	1162	1011
C			443	1860	2012
D				1769	1927
E					283

E. ANALYTICAL COMPARISON OF BATTLE DATA SETS

Having calculated the differences between probability surfaces generated from the empirical distribution functions of random samples of the ratios of fitted coefficients, we now use the 50 samples obtained from the randomization distribution of the surfaces differences and the observed value of our test statistic (T) to compare the battle data sets from the two simulation models being compared. To do so we simply observe the relative location of the observed statistic in the randomization distribution by calculating the percentage of realization values greater then the observed value. This is equivalent to determining the percentile in which it lies. If the percentage is large then we do not reject a hypothesis that the two battle data sets came from similar (possibly the same) simulation models. However, if the percentage is small then we reject that hypothesis. Since our trials resulted in percentages well out in the tails of the randomization distributions (see Table 10), we did not postulate any conclusions with respect to an exact cut off percentage for rejection.

TABLE 10. ANALYTICAL COMPARISONS OF RESULTS (THE NUMBERS INDICATE THE PERCENTAGE OF SAMPLE VALUES OF THE RANDOMIZATION DISTRIBUTION GREATER THAN THE TEST STATISTIC).

	Analytical Comparison				
	B	C	D	E	F
$\nu(A,\_)$	DO NOT REJECT 0.96	REJECT 0.00	REJECT 0.00	REJECT 0.26	REJECT 0.14
$\nu(B,\_)$		REJECT 0.00	REJECT 0.00	REJECT 0.08	REJECT 0.08
$\nu(C,\_)$			DO NOT REJECT 0.74	REJECT 0.10	REJECT 0.04
$\nu(D,\_)$				REJECT 0.04	REJECT 0.02
$\nu(E,\_)$					DO NOT REJECT 0.90

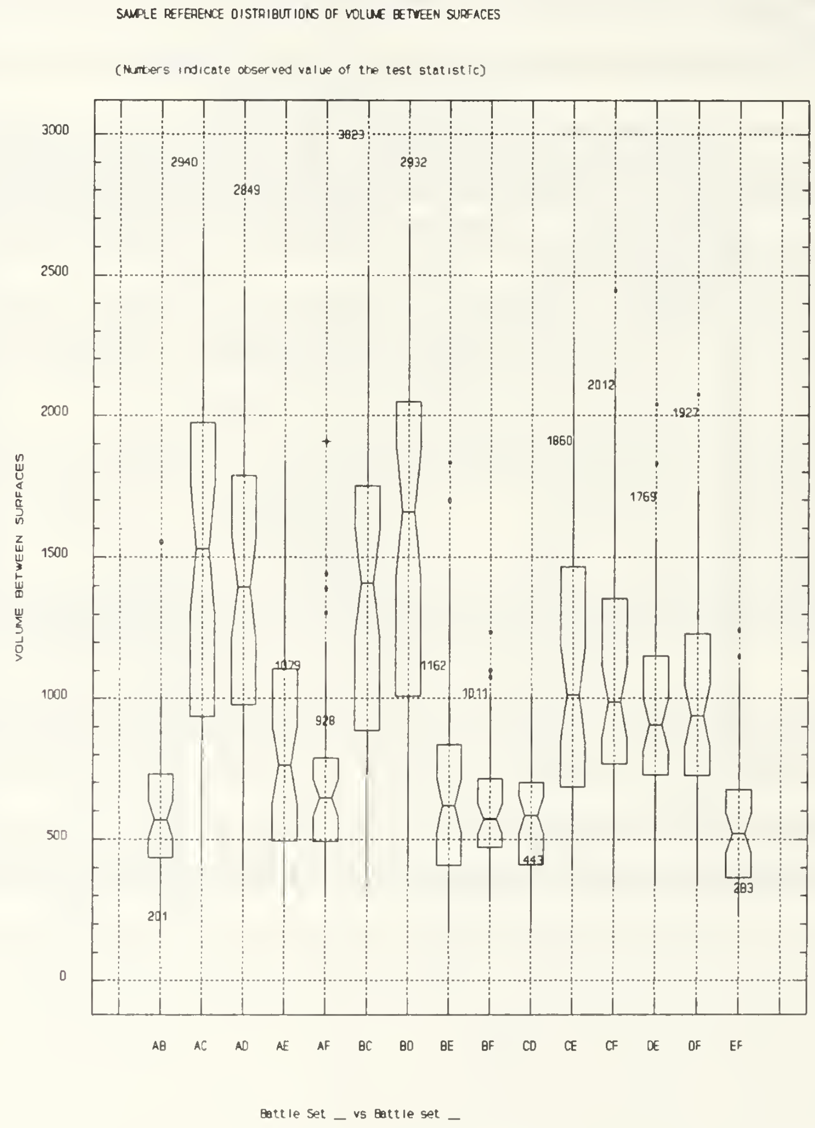

Figure 10. Box Plot Comparison of Surface Differences.

V. CONCLUSIONS AND RECOMMENDATIONS

A. INTRODUCTION

This thesis has presented a method for comparing highly complex combat models using a simple analytical surrogate model. In doing so, we have evaluated a number of fitting techniques and other comparison techniques, and we have developed a discrete analog to Lanchester's aimed fire model.

B. SURROGATE MODELS

We have concentrated here on Lanchester-like models of combat; however, other models could be studied. Non-homogeneous models which include area fire weapons (such as artillery and mortars) were not studied in this thesis but should be studied if this methodology is to be useful as an analytical tool. Models specifically addressing human factors should also be explored.

C. FITTING TECHNIQUES

Although we found that the simultaneous least squares (GAMS model) technique most effectively represented the underlying structure of battle data sets fitted to our model, all of the fitting techniques considered have utility for certain types of problems. Simulated annealing is a very time consuming procedure, but could be useful if the simulation models being studied allow reinforcement. A useful algorithm for fitting non-linear, non-convex continuous functions using simulated annealing is discussed in Brooks and Verdini [Ref. 15].

GAMS provided a very flexible experimental tool for fitting our models. Numerous possible fitting techniques can be written in GAMS in an efficient manner.

We expect that attempts to use multiple regression (as in our proportional independent procedure) with more complex models will experience difficulty with the independence assumption. It has, however, provided a powerful tool for fitting a large number of battles in a very short time.

D. RECOMMENDATION

Comparing high resolution models of combat continues to be a challenging problem. We have seen here that it is feasible to use very simple analytic surrogate models to compare machine generated battle data sets. There is also a lot of work to be done in applying this methodology to actual data sets generated by Janus(A) and at NTC, and in investigating various surrogate models. Successful comparison of these two highly complex models of combat should be of great benefit to the Army analysis community.

APPENDIX A

MATRIX SOLUTION TO DISCRETE MODEL

In Chapter III we developed the 2 x 2 case of the discrete difference equation model. In doing so we combined the fire power from the various weapon systems into one term by using the fire power indices. Suppose instead we wish to consider a model where the attrition of each weapon system and its contribution to the attrition of all other weapon systems are considered separately. The discrete model for this system would be given by:

$$
\begin{aligned}
X_{1,k+1} - X_{1,k} &= -a_{11} Y_{1,k} - a_{12} Y_{2,k} \bullet \bullet \bullet - a_{1m} Y_{m,k} , \\
X_{2,k+1} - X_{2,k} &= -a_{21} Y_{1,k} - a_{22} Y_{2,k} \bullet \bullet \bullet - a_{2m} Y_{m,k} , \\
&\vdots \\
X_{n,k+1} - X_{n,k} &= -a_{n1} Y_{1,k} - a_{n2} Y_{2,k} \bullet \bullet \bullet - a_{nm} Y_{m,k} , \\
Y_{1,k+1} - Y_{1,k} &= -b_{11} X_{1,k} - b_{12} X_{2,k} \bullet \bullet \bullet - b_{1n} X_{n,k} , \\
Y_{2,k+1} - Y_{2,k} &= -b_{21} X_{1,k} - b_{22} X_{2,k} \bullet \bullet \bullet - b_{2n} X_{n,k} , \\
&\vdots \\
Y_{m,k+1} - Y_{m,k} &= -b_{m1} X_{1,k} - b_{m2} X_{2,k} \bullet \bullet \bullet - b_{mn} X_{n,k} .
\end{aligned}
$$

where,

$X_{i,k} \equiv$ the number of X force weapon type i, active in the battle at the end of time increment k,

$Y_{j,k} \equiv$ the number of Y force weapon type j, active in the battle at the end of time increment k,

$a_{j,i} \equiv$ the rate at which Y force weapon type j kills X force weapon type i,

$b_{i,j} \equiv$ the rate at which X force weapon type i kills Y force weapon type j.

or in matrix notation:

$$\begin{bmatrix} X_{1,k+1} \\ X_{2,k+1} \\ \vdots \\ X_{n,k+1} \\ Y_{1,k+1} \\ Y_{2,k+1} \\ \vdots \\ Y_{m,k+1} \end{bmatrix} = \begin{bmatrix} 1 \; 0 \; 0 \cdots 0 & -a_{11} & \cdots & -a_{1m} \\ 0 \; 1 \; 0 \cdots 0 & -a12 & \cdots & -a_{2m} \\ & \vdots & & \\ 0 \cdots 0 \; 1 & -a_{n1} & \cdots & -a_{nm} \\ -b_{11} \cdots -b_{1n} & 1 & 0 \; 0 \cdots & 0 \\ -b21 \cdots -b_{2n} & 0 & 1 \; 0 \cdots & 0 \\ & \vdots & & \\ -b_{m1} \cdots -b_{mn} & 0 & \cdots \; 0 & 1 \end{bmatrix} \begin{bmatrix} X_{1,k} \\ X_{2,k} \\ \vdots \\ X_{n,k} \\ Y_{1,k} \\ Y_{2,k} \\ \vdots \\ Y_{m,k} \end{bmatrix}$$

In block notation we can write:

$$\begin{bmatrix} X_{k+1} \\ Y_{k+1} \end{bmatrix} = \begin{bmatrix} I & A \\ B & I \end{bmatrix} \begin{bmatrix} X_k \\ Y_k \end{bmatrix}.$$

or

$$F_{k+1} = R \, F_k \; ; \quad where \quad R = \begin{bmatrix} I & A \\ B & I \end{bmatrix}.$$

The solution to this system is:

$$F(k) = R^k \, F(0) .$$

Associated with R are its eigenvalues $\{\lambda_1, \lambda_2, \bullet \bullet \bullet, \lambda_{n+m}\}$ and their corresponding eigenvectors $\{\Lambda_1, \Lambda_2, \bullet \bullet \bullet, \Lambda_{n+m}\}$. These eigenvectors form a basis for $\mathbb{R}^{n+m}$ so we can then write F(0) as a linear combination of these eigenvectors:

$$F(0) = c_1 \Lambda_1 + c_2 \Lambda_2 + \cdots + c_{n+m} \Lambda_{n+m} .$$

So finally, the solution to the discrete system can be written:

$$F(k) = c_1 (\lambda_1)^k \Lambda_1 + c_2 (\lambda_2)^k \Lambda_2 + \cdots + c_{n+m} (\lambda_{n+m})^k \Lambda_{n+m} .$$

APPENDIX B

SIMULATED ANNEALING CODE

```
      PROGRAM siman

C** Simulated Annealing Algorithm implemented for the
C** discrete analog to Lanchester's Aimed Fire Model.

      REAL    cred(100),cblue(100),red(100),blue(100),c(4),
     &        acost,ccost,fnacc,fniter,pacc,temp,cool,minpct
      INTEGER n,maxcyc
      COMMON/vars/cred,cblue,red,blue,n

      INTEGER          nacc,niter
      REAL             tcost,mesh(4),a(4),at(4)
      DOUBLE PRECISION dseed

      dseed = 18564245.0d0

       DO 31 i = 1 , 4
         a(i) = 0.0
         c(i) = 0.0
31     CONTINUE

      mesh(1) = 0.0001
      mesh(2) = 0.001
      mesh(3) = 0.0001
      mesh(4) = 0.001

      temp = 100.0
      cool = 0.95
      maxcyc = 100
      minpct = 0.01

      CALL readit
      CALL evalfn(a,acost)
      ccost = acost
```

```
C** Begin Simulated Annealing Algorithm
      DO 36 k = 1 , maxcyc
      nacc  = 0
      niter = 5000
      DO 35 j = 1 , niter
        CALL nbhd(a,at,mesh,dseed)
        CALL evalfn(at,tcost)
        IF (tcost .le. acost) THEN
          DO 33 i = 1 , 4
            a(i) = at(i)
            IF (tcost .lt. ccost)  c(i) = a(i)
33          CONTINUE
          acost = tcost
          IF (acost .lt. ccost) ccost = acost
          nacc = nacc + 1
        ELSE
          CALL unif(dseed,u)
          IF (u .gt. exp(-temp*(tcost-acost))) THEN
            DO 34 i = 1 , 4
              a(i) = at(i)
34            CONTINUE
            acost = tcost
            nacc = nacc + 1
          ENDIF
        ENDIF
35     CONTINUE
      fnacc = FLOAT(nacc)
      fniter = FLOAT(niter)
      pacc = fnacc / fniter
C** Geometric Cooling
      temp = cool * temp
      IF (pacc .lt. minpct) THEN
         PRINT *,’End conditions met in ’,k,’ cycles’
         STOP
       ENDIF
36     CONTINUE
      PRINT *,’Maximum iterations reached’
201    FORMAT(1x,’A:’,4(F12.5),/,1x,’C:’,4(F12.5))
202    FORMAT(1x,’ACOST:’,F12.5,’   CCOST:’,F12.5,
     &’   PCT ACCEPTED:’,F8.3,/)
      END
      SUBROUTINE readit

      REAL    cred(100),cblue(100),red(100),blue(100)
      INTEGER n
      COMMON/vars/cred,cblue,red,blue,n

      INTEGER       ulist,ufpwr,udata,uout,d(5),numobs,i,time
      REAL          fpwr(5)
      CHARACTER*9  bfile
      ulist = 8
      ufpwr = 9
      udata = 10
      uout  = 11
      OPEN(UNIT=ulist,FILE='slist.dat',STATUS='old')
      OPEN(UNIT=ufpwr,FILE='fpwr.dat',STATUS='old')
      DO 1 i = 1 , 5
        READ(ufpwr,101) fpwr(i)
1     CONTINUE
      n = 1
2     CONTINUE
C** Read list of Battle Data Set Files to be used.
      READ(ulist,102,END=4) bfile
      OPEN(UNIT=udata,FILE=bfile,STATUS='old')
        READ(udata,103) numobs
        DO 3 i = n , n+numobs-1
          READ(udata,104) time,d(1),d(2),d(3),d(4),d(5)
          red(i)  = FLOAT(d(1))*fpwr(1) +
FLOAT(d(2))*fpwr(2)
          blue(i) = FLOAT(d(3))*fpwr(3) +
FLOAT(d(4))*fpwr(4) +
     &              FLOAT(d(5))*fpwr(5)
          IF (i .gt . 1) THEN
            cred(i-1)  = red(i)  - red(i-1)
            cblue(i-1) = blue(i) - blue(i-1)
          ENDIF
3        CONTINUE
        n = n+numobs-1
        GOTO 2
4     CONTINUE
```

```
      OPEN(UNIT=uout,FILE='sa.out',STATUS='unknown')
      DO 5 i = 1 , n-1
         WRITE(uout,105) i,red(i),cred(i),blue(i),cblue(i)
5      CONTINUE
      n = n-1
101   FORMAT(F10.7)
102   FORMAT(A9)
103   FORMAT(I5)
104   FORMAT(6I5)
105   FORMAT(I3,5(2x,F10.7))
      RETURN
      END

      SUBROUTINE evalfn(c,ecost)

C** Sum of Square Differences (Linear Lanchester)

      REAL    cred(100),cblue(100),red(100),blue(100)
      INTEGER n
      COMMON/vars/cred,cblue,red,blue,n

      REAL ecost,r,b,c(4)

      ecost = 0.0
      DO 201 i = 1 , n
         r = (cred(i)  - (c(1)*blue(i) + c(2)))**2
         b = (cblue(i) - (c(3)*red(i)  + c(4)))**2
         ecost = ecost + r + b
201   CONTINUE
      ecost = SQRT(ecost)
      RETURN
      END
      SUBROUTINE nbhd(b,bt,m,dseed)

C**   Chooses random neighboring solution.

      REAL            b(4),bt(4),m(4),u,s
      DOUBLE PRECISION dseed

      CALL unif(dseed,u)
      CALL unif(dseed,s)
      j = INT((u*4.0)+0.5)
      sign =  (s-0.5)/abs(s-0.5)
      bt(j) = b(j) + sign*m(j)

      RETURN
      END

      SUBROUTINE UNIF(DSEED,U)
C**   Uniform (0,1) Pseudo Random Number Generator (Lewis &
C**   Orav [Ref. 16])
      INTEGER I
      REAL    U
      DOUBLE PRECISION DENOM,DSEED

      DATA DENOM/2147483647.D0/
      DSEED = DMOD(16807.D0*DSEED,DENOM)
      U = DSEED/DENOM
      RETURN
      END
```

APPENDIX C
STEEPEST DESCENT CODE

```
      PROGRAM steepd

C** Steepest Descent Algorithm

      REAL     cred(100),cblue(100),red(100),blue(100),c(4),
     &         acost,ccost,fnacc,fniter,pacc,temp,cool,minpct
      INTEGER n,maxcyc
      COMMON/vars/cred,cblue,red,blue,n

      INTEGER          nacc,niter,tmax,term
      REAL             tcost,mesh(4),a(4),at(4),amin(4),amax(4)

       DO 31 i = 1 , 4
         a(i) = 0.0
         c(i) = 0.0
31     CONTINUE

      mesh(1) = 0.0001
      mesh(2) = 0.001
      mesh(3) = 0.0001
      mesh(4) = 0.001
      amin(1) = -1.0
      amax(1) =  1.0
      amin(2) = -5.0
      amax(2) =  5.0
      amin(3) = -1.0
      amax(3) =  1.0
      amin(4) = -5.0
      amax(4) =  5.0

      precis = 0.0001
      refine = 0.60
      maxcyc = 20
      maxit  = 2500
      CALL readit
      CALL evalfn(a,acost)
      pcost = acost
      ncyc = 1
      it = 1

C Top of while loop ..................
C** Finds Direction of greatest improvement in the Cost Fcn.
1000  delmax = 0.0d0
      DO 1001 term = 1 , 4
        a(term) = a(term) + mesh(term)
        IF (a(term) .le. amax(term)) THEN
          CALL evalfn(a,tcost)
          delta = acost - tcost
          If (delta .gt. delmax) THEN
            delmax = delta
            tmax = term
            sign = 1.0d0
          ENDIF
        ENDIF
          a(term) = a(term) - 2.0d0*mesh(term)
          IF (a(term) .ge. amin(term)) THEN
            CALL evalfn(a,tcost)
            delta = acost - tcost
            IF (delta .gt. delmax) THEN
              delmax = delta
              tmax = term
              sign = -1.0d0
            ENDIF
          ENDIF
          a(term) = a(term) + mesh(term)
1001  CONTINUE
C** If an improving direction is found move in that Direction
C** on grid [mesh] square.
      IF ((delmax .gt. 0.0) .and. (it .le. maxit)) THEN
        a(tmax) = a(tmax) + sign*mesh(tmax)
        CALL evalfn(a,acost)
        it = it + 1
        GOTO 1000
C** If change in cost during this cycle less then set Precision
C** We are done!
```

```
      ELSEIF ((pcost-acost) .lt. precis) THEN
        GOTO 1003
C** If num of cycles is < maximum cycle limit => Refine Grid        ELSEIF (ncyc
.lt. maxcyc) THEN
        ncyc = ncyc +1
        pcost = acost
        DO 1002 term = 1, 4
          mesh(term) = mesh(term) * refine
1002      CONTINUE
        it = 1
        GOTO 1000
      ENDIF
1003  CONTINUE
      Print *,'ncyc,acost, ',ncyc,acost,it,maxit
      Print *,'a',a
      PRINT *,'end'
      WRITE(*,392) ' COST',FLOAT(acost)

390   FORMAT(1x,A6,8x,A12)
391   FORMAT(1x,A25,2x,F10.7)
392   FORMAT(1x,A5,10x,F12.7)

36    CONTINUE
      END


      SUBROUTINE readit

      REAL    cred(100),cblue(100),red(100),blue(100)
      INTEGER n
      COMMON/vars/cred,cblue,red,blue,n

      INTEGER      ulist,ufpwr,udata,uout,d(5),numobs,i,time
      REAL         fpwr(5)
      CHARACTER*9  bfile

      ulist  = 8
      ufpwr  = 9
      udata  = 10
      uout   = 11

      OPEN(UNIT=ulist,FILE='slist.dat',STATUS='old')
      OPEN(UNIT=ufpwr,FILE='fpwr.dat',STATUS='old')
      DO 1 i = 1 , 5
        READ(ufpwr,101) fpwr(i)
1     CONTINUE
      n = 1
2     CONTINUE
      READ(ulist,102,END=4) bfile
      OPEN(UNIT=udata,FILE=bfile,STATUS='old')
        READ(udata,103) numobs
        DO 3 i = n , n+numobs-1
          READ(udata,104) time,d(1),d(2),d(3),d(4),d(5)
          red(i)  = FLOAT(d(1))*fpwr(1) +
FLOAT(d(2))*fpwr(2)
          blue(i) = FLOAT(d(3))*fpwr(3) +
FLOAT(d(4))*fpwr(4) +
     &              FLOAT(d(5))*fpwr(5)
          IF (i .gt . 1) THEN
            cred(i-1)  = red(i)  - red(i-1)
            cblue(i-1) = blue(i) - blue(i-1)
          ENDIF
3       CONTINUE
        n = n+numobs-1
        GOTO 2
4     CONTINUE
      OPEN(UNIT=uout,FILE='sa.out',STATUS='unknown')
      DO 5 i = 1 , n-1
        WRITE(uout,105) i,red(i),cred(i),blue(i),cblue(i)
5     CONTINUE
      n = n-1
101    FORMAT(F10.7)
102    FORMAT(A9)
103    FORMAT(I5)
104    FORMAT(6I5)
105    FORMAT(I3,5(2x,F10.7))
      RETURN
      END
```

```
   SUBROUTINE evalfn(c,ecost)

C** Sum of Square Differences (Linear Lanchester Model)

      REAL    cred(100),cblue(100),red(100),blue(100)
      INTEGER n
      COMMON/vars/cred,cblue,red,blue,n

      REAL ecost,r,b,c(4)

      ecost = 0.0
      DO 201 i = 1 , n
        r = (cred(i)  - (c(1)*blue(i) + c(2)))**2
        b = (cblue(i) - (c(3)*red(i)  + c(4)))**2
        ecost = ecost + r + b
201   CONTINUE
      ecost = SQRT(ecost)
      RETURN
      END
```

APPENDIX D
GAMS REGRESSION MODEL

```
$TITLE Least Squares Fit of Battle Data to Analytical Model
$STITLE (Linear Lanchester Form - 2-stage fit)

*---------- GAMS OPTIONS AND DOLLAR CONTROL OPTIONS ----------

$OFFUPPER OFFSYMXREF OFFSYMLIST

 OPTIONS LIMCOL = 0, LIMROW = 0, SOLPRINT = OFF
 OPTIONS RESLIM = 50, ITERLIM = 10000, OPTCR = 0.0;
*----------- DEFINITIONS AND DATA --------------------------

 SETS
     I    increments      /1* 12/

     W    weapons      /
                  RTANK
                  RAPC
                  BTANK
                  BAPC
                  BTOW /;

 PARAMETER FP(W) fire power /
                     RTANK    .920
                     RAPC     .780
                     BTANK   1.000
                     BAPC     .850
                     BTOW     .750 /;
```

```
TABLE CHANGE(I,W)  changes in wpn level in inc

          RTANK  RAPC   BTANK  BAPC   BTOW
   1        7     8     4     4     4
   2        8     8     3     4     4
   3        6     8     4     3     4
   4        6     7     3     3     3
   5        6     6     2     3     3
   6        6     6     3     3     3
   7        5     6     2     2     2
   8        5     6     2     2     3
   9        4     5     2     2     2
  10        5     5     2     2     2
  11        4     5     1     2     1
  12        4     4     1     1     2 ;

TABLE LEVEL(I,W)   level of wpn at start of inc

          RTANK  RAPC   BTANK  BAPC   BTOW
   1       80    160     60   120     40
   2       73    152     56   116     36
   3       65    144     53   112     32
   4       59    136     49   109     28
   5       53    129     46   106     25
   6       47    123     44   103     22
   7       41    117     41   100     19
   8       36    111     39    98     17
   9       31    105     37    96     14
  10       27    100     35    94     12
  11       22     95     33    92     10
  12       18     90     32    90      9 ;

 PARAMETER RED(I)  red firepower at increment i;

         RED(I)  = FP("RTANK") * LEVEL(I,"RTANK")
              + FP("RAPC")  * LEVEL(I,"RAPC");

PARAMETER CRED(I) change in red firepower in increment;

         CRED(I) = FP("RTANK") * CHANGE(I,"RTANK")
              + FP("RAPC")  * CHANGE(I,"RAPC");
 PARAMETER BLUE(I) blue firepower at increment i;

        BLUE(I) = FP("BTANK") * LEVEL(I,"BTANK")
               + FP("BAPC")  * LEVEL(I,"BAPC")
               + FP("BTOW")  * LEVEL(I,"BTOW");

 PARAMETER CBLUE(I)  change in firepower of blue in increment;

        CBLUE(I)= FP("BTANK") * CHANGE(I,"BTANK")
               + FP("BAPC")  * CHANGE(I,"BAPC")
               + FP("BTOW")  * CHANGE(I,"BTOW");

*----------MODEL----------
 POSITIVE VARIABLES
    A           coef on blue level in equation red
    C           coef on red  level in equation blue
    BL(I)        residual in blue equation
    RD(I)        residual in red  equation;

 VARIABLES
    B           additive noise term
    D           additive noise term;

 VARIABLE
    COST          sum of squared error;

 EQUATIONS
    OBJ          minimize sum of squared error
    DRED1(I)       sq error increment no noise term
    DBLUE1(I)      sq error increment no noise term
    DRED2(I)       sq error in increment with noise term
    DBLUE2(I)      sq error in increment with noise term;

*       >>>>>>>>>>  MINIMIZE  <<<<<<<<<<
 OBJ..  COST =E= SUM(I, RD(I)) + SUM(I, BL(I)) ;

*       >>>>>>>>>> SUBJECT TO <<<<<<<<<<

 DRED1(I)..   RD(I) =G= POWER((CRED(I)  - ((A*BLUE(I)+B))),2);
 DRED2(I)..   RD(I) =G= POWER((CRED(I)  - ((A*BLUE(I)+B))),2);
 DBLUE1(I)..  BL(I) =G= POWER((CBLUE(I) - ((C*RED(I) +D))),2);
 DBLUE2(I)..  BL(I) =G= POWER((CBLUE(I) - ((C*RED(I) +D))),2);
 MODEL LSQ1 /OBJ,DRED1,DBLUE1/;
 SOLVE LSQ1 USING NLP MINIMIZING COST;

 A.FX = A.L;
 C.FX = C.L;

 MODEL LSQ2 /OBJ,DRED2,DBLUE2/;
 SOLVE LSQ2 USING NLP MINIMIZING COST;

*----------REPORTS----------
 DISPLAY COST.L,A.L,B.L,C.L,D.L
```

APPENDIX E

PROBABILITY SURFACE COMPARISON CODE

```
      PROGRAM datmat

      INTEGER first(2),next(2,100)
      REAL    ecdf(2,100)
      COMMON/list/first,next,ecdf


      INTEGER          r1(20),r2(20),lower,higher
      REAL            dmatrx(6,6,500),cdfdat(6,20),r(40)
      REAL            plow,phigh,tdiff(6,6)

      CHARACTER*12      cfile
      DOUBLE PRECISION dseed


C      dseed = 7561883.d0
       dseed = 829847.d0
      OPEN(UNIT=8,FILE='cdfile.dat',STATUS='old')
      n = 0
111    CONTINUE
        n = n + 1
        READ(8,211,END=55) cfile
        OPEN(UNIT=9,FILE=cfile,STATUS='old')
        m = 0
112      CONTINUE
          m = m + 1
          READ(9,212,END=56) cdfdat(n,m)
          GOTO 112
56       CONTINUE
        CLOSE(9)
        GOTO 111
55    CONTINUE
      CLOSE(8)
      OPEN(UNIT=10,FILE='datmat.out',STATUS='unknown')
      OPEN(UNIT=11,FILE='pdat.out',STATUS='unknown')
      DO 113 i = 1 , 5
        DO 114 j = i+1 , 6
          lower = 0
          higher = 0
          PRINT*, 'Comparing surfaces for : ',i,' vs ',j
          DO 110 l = 1 , 20
            ecdf(1,l) = cdfdat(i,l)
            ecdf(2,l) = cdfdat(j,l)
110         CONTINUE
          CALL tcdf(tdiff(i,j))
          WRITE(11,*) 'i,j',i,j,' tdiff = ',tdiff(i,j)
          PRINT *,'tdiff = ',tdiff(i,j)
          DO 99 l = 1 , 20
            r(l)    = cdfdat(i,l)
            r(l+20) = cdfdat(j,l)
99          CONTINUE
          DO 116 l = 1 , 50
            CALL roll(20,40,r1,r2,dseed)
            DO 115 k = 1 , 20
              ecdf(1,k) = r(r1(k))
              ecdf(2,k) = r(r2(k))
115           CONTINUE
            CALL tcdf(dmatrx(i,j,l))
            IF(dmatrx(i,j,l) .LT. tdiff(i,j)) lower = lower +1
            IF(dmatrx(i,j,l) .GT. tdiff(i,j)) higher= higher+1
116         CONTINUE
          plow = FLOAT(lower)/50.0
          phigh = FLOAT(higher)/50.0
114       CONTINUE
113     CONTINUE

      DO 117 i = 1 , 5
        DO 118 j = i+1 , 6
          DO 119 k = 1 , 50
            WRITE(10,213) dmatrx(i,j,k)
119         CONTINUE
118       CONTINUE
117     CONTINUE
211     FORMAT(A12)
212     FORMAT(F10.7)
213     FORMAT(F10.4)

      END
      SUBROUTINE  tcdf(diff)

      INTEGER first(2),next(2,100)
      REAL    ecdf(2,100)
      COMMON/list/first,next,ecdf

      REAL         deltax,deltay,x,y,diff,ratio

      DO 33 nc = 1 ,2
        DO 32 n = 1 , 10
          next(nc,n) = 0
          CALL ordcdf(nc,n)
32       CONTINUE
33    CONTINUE
      diff = 0.0
      deltax = 1.0
      deltay = 1.0
      DO 21 x = 50, 200, deltax
        Do 22 y = 50, 200, deltay
          ratio = (x/y)**2
          CALL cdf(1,ratio,prob1)
          CALL cdf(2,ratio,prob2)
          diff = diff + (abs(prob1-prob2) * (deltax*deltay))
22      CONTINUE
21    CONTINUE

100   FORMAT(F10.7)

      END

      SUBROUTINE ordcdf(nc,n)

      INTEGER count,test,ltest,ttest,maxcnt

      INTEGER first(2),next(2,100)
      REAL    ecdf(2,100)
      COMMON/list/first,next,ecdf

      maxcnt = 100
      IF (n .EQ. 1) THEN
        first(nc)   = 1
        next(nc,1) = 0
        count    = 1
        RETURN
      ELSEIF (count .EQ. maxcnt) THEN
        PRINT *, 'OVERFLOW ERROR >>> CDF LIST IS FULL <<<'
        STOP
      ELSE
        count = count + 1
        test  = first(nc)
201      CONTINUE
          IF (ecdf(nc,n) .LT. ecdf(nc,test)) THEN
            IF (test .EQ. first(nc)) THEN
              first(nc) = n
            ELSE
             next(nc,ltest) = n
            ENDIF
            next(nc,n)     = test
            RETURN
          ELSE
            ttest = next(nc,test)
            IF (ttest .EQ. 0) THEN
              next(nc,n)     = 0
              next(nc,test) = n
              RETURN
            ELSE
              ltest = test
              test  = ttest
              GOTO 201
            ENDIF
          ENDIF
      ENDIF
      RETURN

      END
      SUBROUTINE cdf(nc,pt,prob)

      INTEGER first(2),next(2,100)
      REAL    ecdf(2,100)
      COMMON/list/first,next,ecdf

      INTEGER nc,test
      REAL    pt,prob

      num = 0
      test = first(nc)
400    CONTINUE
      IF (pt .LE. ecdf(nc,test)) THEN
         prob = FLOAT(num)/10.0
         RETURN
      ELSE
         num = num + 1
         test = next(nc,test)
         IF (test .EQ. 0) THEN
            prob = 1.0
            RETURN
         ENDIF
      ENDIF
      GOTO 400
      END
      SUBROUTINE roll(n,m,r,rc,dseed)

      INTEGER r(n),rc(n)
      DOUBLE PRECISION dseed

         DO 2 i = 1 , n
            r(i) = 0
3           CONTINUE
            CALL unif(dseed,u)
            num = INT((u*m)+.5)
            DO 1 j = 1 , i
               IF (r(j) .EQ. num) GOTO 3
1           CONTINUE
            r(i) = num
2        CONTINUE
```

```
      k = 1
      DO 6 i = 1 , m
        DO 4 j = 1 , n
          IF (r(j) .EQ. i) GOTO 5
4         CONTINUE
        rc(k) = i
        k = k + 1
5         CONTINUE
6     CONTINUE
      RETURN
      END

      SUBROUTINE UNIF(DSEED,U)
C
      INTEGER I
      REAL    U
      DOUBLE PRECISION DENOM,DSEED
C
      DATA DENOM/2147483647.D0/
C
      DSEED = DMOD(16807.D0*DSEED,DENOM)
      U = DSEED/DENOM
C
      RETURN
      END
```

LIST OF REFERENCES

1. Box, G.E.P., Hunter, W.G., and Hunter, J.S., *Statistics for Experimenters: An introduction to Design, Data Analysis, and Model Building*, pp. 93-106, John Wiley & Sons, Inc., 1978.

2. *JANUS(A) Basic User's Tutorial*, TRAC-RDM-TD 90-1 (DRAFT), US Army TRADOC Analysis Command - Monterey, June 1990.

3. Ingber, L., *Mathematical Comparison of the Computer Models to Exercise Data: Comparison of JANUS(A) to National Training Center Data*, Proposal to: US Army TRADOC Analysis Command, Fort Leavenworth, KS, 29 September 1988.

4. Shy, J., "Jomini," ed. Paret, P., *Makers of Modern Strategy*, pp. 143-185, Princeton University Press, 1986.

5. Taylor, J. G., *Lanchester Models of Warfare, Volume I, p. 52*, Operations Research Society of America, 1983.

6. Taylor, J. G., *Lanchester Models of Warfare, Volume I, p. 53*, Operations Research Society of America, 1983.

7. Sandefur, J. T., *Discrete Dynamical Systems: Theory and Applications*, pp. 213-222, Clarendon Press, 1990.

8. Hartman, J. K., "Lecture Notes in Aggregated Combat Modelling,", Naval Postgraduate School, 1985.

9. Bitters, D., "The Economics of the Battlefield," paper presented at the Naval Postgraduate School, Monterey, CA, November 1990.

10. Barr, D., Weir, M. D., and Hoffman, J., "Evaluation of Combat," Naval Postgraduate School, Monterey, CA, unpublished.

11. Metropolis, W., and others, "Equation of State Calculations by Fast Computing Machines," *Journal of Chemical Physics*, v. 21, pp. 1087-1092, 1953.

12. Kirkpatrick, S., Gelatt, C.D., and Vecchi, M. P., "Optimization by Simulated Annealing," *Science*, v. 220, 13 May 1983.

13. Johnson, D. S., and others, "Optimization by Simulated Annealing: An Experimental Evaluation; Part I, Graph Partitioning," *Operations Research*, v. 37:6, p. 868, November-December 1989.

14. Johnson, D. S., and others, "Optimization by Simulated Annealing: An Experimental Evaluation; Part III, The Traveling Salesman Problem," (in preparation).

15. Brooks, D. G., and Verdini, W. A., "Computational Experience with Generalized Simulated Annealing Over Continuous Variables," *American Journal of Mathematical and Management Sciences*, v. 8:3 & 4, 1988.

16. Lewis, P. A. W., Orav, E. J., *Simulation Methodology for Statistician, Operations Analysts, and Engineers, Volume I*, Wadsworth & Brooks/Cole Advanced Books and Software, 1989.

BIBLIOGRAPHY

Aarts, E., and Korst, J., *Simulated Annealing and Boltzmann Machines*, John Wiley & Sons, Inc., 1989.

Bowman, M., *Integration of the NTC Tactical Database and JANUS(T) Towards a Combat Decision Support System*, Master's Thesis, Naval Postgraduate School, Monterey, CA, 1989.

Brooke, A., Kendrick, D., and Meeraus, A., *Gams: A Users Guide*, The Scientific Press, 1988.

Giordano, F. R., and Weir, M. D., *A First Course in Mathematical Modeling*, Brooks/Cole Publishing Company, 1985.

Golub, G. H., and Van Loan, C. F., *Matrix Computations*, 2d ed., The John Hopkins University Press, 1989.

Ingber, L., "Very Fast Simulated Reannealing,", *Mathematical Computer Modelling*, v. 12, 1989.

Johnson, D. S., and others, "Optimization by Simulated Annealing: An Experimental Evaluation; Part II, Graph Coloring and Number Partitioning," *Operations Research*, (to appear).

Kirkpatrick, S., "Optimization by Simulated Annealing: Quantitative Studies," *Journal of Statistical Physics*, 1984.

Naval Postgraduate School Report 55-79-029, *Calculation of State Probabilities for a Stochastic Lanchester Combat Model*, by L. Billard, November 1979.

Taylor, J. G., *Lanchester Models of Warfare, Volume II*, Operations Research Society of America, 1983.

Tovey, C., "Simulated Simulated Annealing," , *American Journal of Mathematical and Management Sciences*, v. 8:3 & 4, 1988.

Upton, S. C., *Statistical Mechanics Model of Combat*, Master's Thesis, Naval Postgraduate School, Monterey, CA, March 1987.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2. Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3. Professor Donald Barr, Code MA/BA Naval Postgraduate School Monterey, CA 93943-5004	2
4. Professor Patricia Jacobs, Code OR/JC Naval Postgraduate School Monterey, CA 93943-5004	1
5. Professor Maurice D. Weir, Code MA/WC Naval Postgraduate School Monterey, CA 93943-5004	1
6. Colonel Frank R. Giordano Department of Mathematical Sciences United States Military Academy West Point, NY 10996	1
7. Major James Hoffman TRAC-Monterey Monterey, CA 93943-0692	1
8. Captain John R. Green Department of Mathematical Sciences United States Military Academy West Point, NY 10996	3
9. Chairman, Mathematic Department, Code MA Naval Postgraduate School Monterey, CA 93943-5004	1

10. Chairman, Operations Research Department, Code OR 1
Naval Postgraduate School
Monterey, CA 93943-5004

Thesis
G728 Green
c.1 Comparing combat models using analytical surrogates.

Thesis
G728 Green
c.1 Comparing combat models using analytical surrogates.

DUDLEY KNOX LIBRARY
3 2768 00016428 9